

A.A 2020/21

Sistema di Planning Aziendale

Progettazione e sviluppo di una base di dati relazionale in PostgreSQL

di Fascia Fabio (N86003288) e Conte Salvatore (N86003295)

Codice gruppo : 2 - Traccia : 2

Indice

[1. Introduzione 4](#_Toc65258392)

[1.1. Descrizione del problema 4](#_Toc65258393)

[2. Progettazione Concettuale 5](#_Toc65258394)

[2.1. Class Diagram 6](#_Toc65258395)

[2.2. Ristrutturazione del Class Diagram 7](#_Toc65258396)

[2.2.1. Analisi delle chiavi 7](#_Toc65258397)

[2.2.2. Analisi degli attributi derivati 7](#_Toc65258398)

[2.2.3. Analisi delle ridondanze 8](#_Toc65258399)

[2.2.4. Analisi degli attributi strutturati 8](#_Toc65258400)

[2.2.5. Analisi degli attributi a valore multiplo 9](#_Toc65258401)

[2.2.6. Analisi delle gerarchie di specializzazione 9](#_Toc65258402)

[2.3. Class Diagram Ristrutturato 10](#_Toc65258403)

[2.4. Dizionario delle Classi 11](#_Toc65258404)

[2.5. Dizionario delle Associazioni 13](#_Toc65258405)

[2.6. Dizionario dei Vincoli 14](#_Toc65258406)

[3. Progettazione Logica 15](#_Toc65258407)

[3.1. Schema Logico 15](#_Toc65258408)

[4. Progettazione Fisica 17](#_Toc65258409)

[4.1. Definizione Tabelle 17](#_Toc65258410)

[4.1.1. Definizione della Tabella DIPENDENTE 17](#_Toc65258411)

[4.1.2. Definizione della Tabella PROGETTO 17](#_Toc65258412)

[4.1.3. Definizione della Tabella AMBITO 18](#_Toc65258413)

[4.1.4. Definizione della Tabella PARTECIPANTE 18](#_Toc65258414)

[4.1.5. Definizione della Tabella SALA 19](#_Toc65258415)

[4.1.6. Definizione della Tabella MEETINGF 19](#_Toc65258416)

[4.1.7. Definizione della Tabella MEETINGT 20](#_Toc65258417)

[4.1.8. Definizione della Tabella PARTECIPAMF 21](#_Toc65258418)

[4.1.9. Definizione della Tabella PARTECIPAMT 22](#_Toc65258419)

Indice

1. Introduzione
   1. Descrizione del problema
2. Progettazione Concettuale
   1. Class Diagram
   2. Ristrutturazione del Class Diagram
      1. Analisi delle chiavi
      2. Analisi degli attributi derivati
      3. Analisi delle ridondanze
      4. Analisi degli attributi strutturati
      5. Analisi degli attributi a valore multiplo
      6. Analisi delle gerarchie di specializzazione
      7. Class Diagram ristrutturato
   3. Dizionario delle Classi
   4. Dizionario delle Associazioni
   5. Dizionario dei Vincoli
3. Progettazione Logica
   1. Schema logico
4. Progettazione Fisica
   1. Definizione tabelle
      1. Tabella DIPENDENTE
      2. Tabella PROGETTO
      3. Tabella AMBITO
      4. Tabella PARTECIPANTE
      5. Tabella SALA
      6. Tabella MEETINGF
      7. Tabella MEETINGT
      8. Tabella PARTECIPAMF
      9. Tabella PARTECIPAMT
   2. Funzioni, Procedure ed altre Automazioni
      1. Inserimento di Progetti e/o Meeting
      2. Calcolo automatico della valutazione aziendale
5. Esempi d’uso
   1. Popolamento con dati esemplificativi
   2. Creazione di un nuovo progetto
   3. Creazione di un nuovo meeting fisico

# Introduzione

Il seguente elaborato ha lo scopo di documentare la progettazione e lo sviluppo di una base di dati relazionale del DBMS PostgreSQL, ad opera degli studenti Fascia Fabio e Conte Salvatore del CdL in Informatica, presso l’Università degli Studi di Napoli “Federico II”. Il database nasce come progetto a scopi valutativi per il corso di Basi di Dati, ed implementa un sistema di planning aziendale.

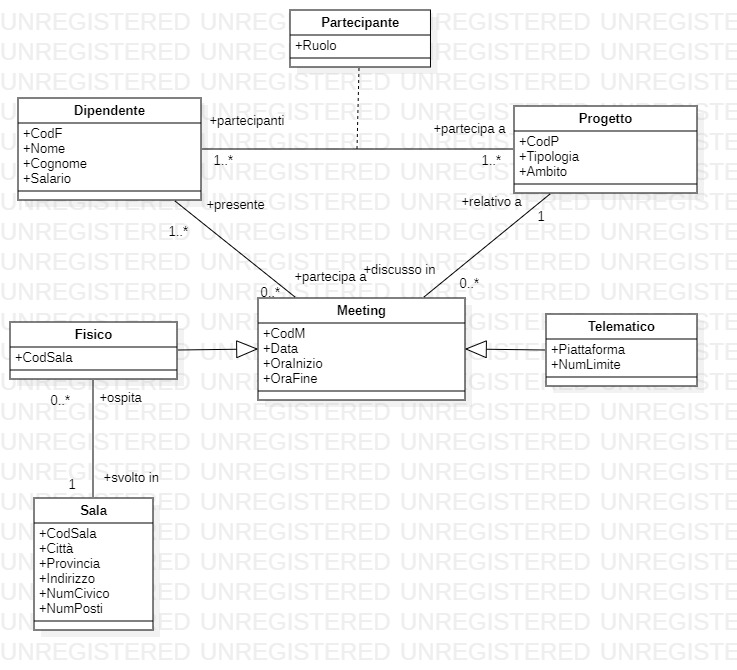
## Descrizione del problema

Verranno riportate progettazione e sviluppo di una base di dati relazionale, che implementi un sistema di planning aziendale che permetta l’organizzazione e la storicizzazione di progetti aziendali. Il sistema tiene traccia dei partecipanti ai progetti, identificando ruoli per ognuno di essi. Ad ogni progetto sono associati una tipologia (ad es. “Ricerca di base”, “Ricerca Industriale”, “Ricerca Sperimentale” ecc.) ed uno o più ambiti (Economia, Medicina, Biologia…). Il sistema permette inoltre l’organizzazione di meeting fisicamente, in sale riunioni, o telematicamente, su una piattaforma di videoconferenza. Viene tenuta traccia delle partecipazioni a progetti e meeting ai fini della valutazione dei singoli partecipanti, gestita con un sistema a punti (ogni partecip azione vale 1 punto nella valutazione).

# Progettazione Concettuale

In questo capitolo documentiamo la progettazione del database al suo livello di astrazione più alto. Partendo dall’analisi dei requisiti da soddisfare, si arriverà ad uno schema concettuale indipendente dalla struttura dei dati e dall’implementazione fisica degli stessi, rappresentato con un Class Diagram UML. Quest’ultimo evidenzierà le entità rilevanti nel problema, oltre alle relazioni che intercorrono tra esse e gli eventuali vincoli da imporre.

## Class Diagram



## Ristrutturazione del Class Diagram

Si procede alla ristrutturazione del Class Diagram, al fine di rendere quest’ultimo idoneo alla traduzione in schemi relazionali e di migliorarne l’efficienza. La ristrutturazione procederà secondo i seguenti punti:

* Analisi delle chiavi
* Analisi degli attributi derivati
* Analisi delle ridondanze
* Analisi degli attributi strutturati
* Analisi degli attributi a valore multiplo
* Analisi delle gerarchie di specializzazione

### Analisi delle chiavi

Ai fini dell’efficienza nella rappresentazione delle varie entità, nello specifico Progetti, Meeting e Sale Riunioni, risulta conveniente l’introduzione di chiavi primarie “surrogate” ovvero rappresentate non da informazioni proprie della singola entità, ma da identificativi di tipo intero associati a ciascuna istanza delle stesse. Ciò permetterà di identificare in maniera computazionalmente meno dispendiosa le varie istanze delle suddette entità.

### Analisi degli attributi derivati

Per ottimizzare ulteriormente l’utilizzo delle risorse di calcolo, analizziamo gli eventuali attributi derivati, ovvero calcolabili da altri attributi delle entità.

Il più evidente risulta essere in questo caso la Valutazione aziendale del singolo Dipendente, calcolabile a partire palle partecipazioni storicizzate di quest’ultimo a Progetti e Meeting. Ad una attenta analisi risulta essere conveniente la storicizzazione di tale valore come attributo del Dipendente stesso, ai fini di una più efficiente ricerca dei Dipendenti in funzione della loro Valutazione.

### Analisi delle ridondanze

Analizziamo ora l’eventuale presenza di associazioni ridondanti tra le varie entità, in maniera tale da evitare incoerenze logiche nella rappresentazione logica dei dati.

L’unico caso riscontrabile è dato dall’associazione tra Progetto e Dipendente: ogni progetto può avere infatti un qualunque numero di partecipanti, ma richiede sempre la presenza di uno e un solo Project Manager. Quest’ultimo è però a sua volta un partecipante al Progetto, motivo per cui risulta logicamente più corretto gestire l’assciazione relativa al Project Manager, a questo punto evidentemente ridondante, con un serie di vincoli interrelazionali, nello specifico uno di unicità ed uno di totalità del PM.

### Analisi degli attributi strutturati

Vanno ora analizzati e concettualmente corretti eventuali attributi strutturati presenti nelle entità. Questi infatti non sono logicamente rappresentabili all’interno di un DBMS, e vanno quindi eliminati e codificati in altro modo.

Fortunatamente, nella rappresentazione concettuale presentata non sono presenti attributi strutturati.

### Analisi degli attributi a valore multiplo

Verifichiamo ora la presenza di eventuali attributi a valore multiplo, anch’essi non logicamente rappresentabili e quindi da eliminare nello schema concettuale ristrutturato.

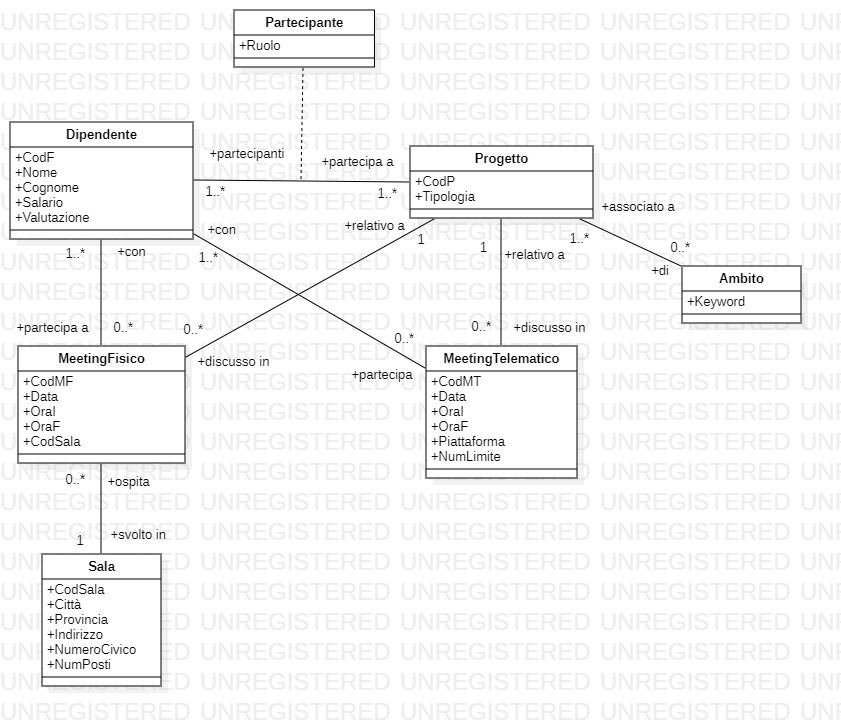
Un attributo a valore multiplo è l’Ambito di un progetto: un singolo progetto può infatti avere più di un ambito. Per correggere tale problema logico, rappresentiamo l’attributo tramite una nuova entità Ambito, associata uno a molti con i Progetti.

### Analisi delle gerarchie di specializzazione

Infine andiamo a ristrutturare eventuali gerarchie di specializzazione, altro elemento non rappresentabile in un DBMS relazionale.

L’unica gerarchia di specializzazione nel Class Diagram presentato è quella riguardante i Meeting: il generico meeting si specializza infatti in Meeting Fisico oppure in Meeting Telematico. Tale specializzazione è Totale (ogni Meeting deve essere specializzato) e Disgiunta (un Meeting non può essere sia Fisico che Telematico), dunque risulta conveniente ai fini della rappresentazione logica ricontestualizzare tale specializzazione tramite un “appiattimento”: la generalizzazione Meeting viene eliminata e cede i suoi attributi e le sue associazioni alle singole specializzazioni. Il risultato saranno quindi due entità, Meeting Fisico e Meeting Telematico, indipendenti l’una dall’altra.

## Class Diagram Ristrutturato



## Dizionario delle Classi

|  |  |  |
| --- | --- | --- |
| Classe | Descrizione | Attributi |
| Dipendente | Descrive ciascun dipendente dell’azienda fruitrice del sistema. | * **CodF** (char[16]) **:** Codice Fiscale del Dipendente. * **Nome** (string): Nome del Dipendente. * **Cognome** (string) : Cognome del Dipendente. * **Salario** (float) : Salario attuale del Dipendente. * **Valutazione** (int) : Valutazione aziendale del Dipendente, espressa come punteggio in numeri interi. |
| Progetto | Descrive i singoli progetti portati avanti dall’azienda. | * **CodP** (int) : Codice identificativo univoco del Progetto (chiave surrogata). * **Tipologia** (string) : Tipologia del progetto. * **Descrizione** (string, parziale) : Breve descrizione del progetto. |
| Ambito | Codifica dell’attributo a valore multiplo di Progetto: descrive i singoli ambiti di un progetto. | * **Keyword** (string) : Nome dell’ambito. |
| Meeting Fisico | Descrive un Meeting Fisico tenutosi per un Progetto. | * **CodMF** (int) : Codice identificativo univoco del Meeting Fisico (chiave surrogata). * **Data** (date) : Data in cui il Meeting si è tenuto. * **OraI** (time) : Ora di inizio del Meeting. * **OraF** (time) : Ora in cui il Meeting è terminato. |

|  |  |  |
| --- | --- | --- |
| Classe | Descrizione | Attributi |
| Meeting Telematico | Descrive un Meeting Telematico tenutosi per un Progetto. | * **CodMT** (int) : Codice identificativo univoco del Meeting Telematico (chiave surrogata). * **Data** (date) : Data in cui il Meeting si è tenuto. * **OraI** (time) : Ora di inizio del Meeting. * **OraF** (time) : Ora in cui il Meeting è terminato. * **Piattaforma** (string) : Piattaforma di videoconferenze su cui il Meeting si è svolto. * **NumLimite** (int, parziale) : Eventuale limite al numero di partecipanti al Meeting Telematico, dettato dalla Piattaforma oppure dagli organizzatori. |
| Sala | Descrive una sala riunioni in cui è possibile organizzare Meeting Fisici. | * **CodSala** (int) : Codice identificativo univoco della Sala riunioni (chiave surrogata). * **Città** (string) : Città in cui è presente la Sala riunioni. * **Provincia** (char[2]) : Provincia della Città specificata. * **Indirizzo** (string) : Indirizzo della Sala riunioni. * **NumeroCivico** (int) : Numero Civico della Sala riunioni. * **NumPosti** (int) : Numero dei posti disponibili nella Sala. |

## Dizionario delle Associazioni

|  |  |  |
| --- | --- | --- |
| Nome | Descrizione | Classi coinvolte |
| Partecipante | Esprime la partecipazione di un Dipendente ad un determinato Progetto. | **Dipendente** [1..\*] ruolo partecipante : indica un Dipendente che partecipa ad un certo Progetto.  **Progetto** [1..\*] ruolo partecipa a : indica il progetto a cui un certo dipendente partecipa.  **Ruolo** classe di associazione : indica il ruolo che un determinato Dipendente svolge in un determinato Progetto. |
| Partecipante Meeting Fisico | Esprime la partecipazione di un Dipendente ad un Meeting Fisico. | **Dipendente** [1..\*] ruolo partecipante : indica un Dipendente che partecipa ad un certo Meeting Fisico.  **Meeting Fisico** [0..\*] ruolo partecipa a : indica un Meeting Fisico a cui partecipa un determinato Dipendente. |
| Partecipante Meeting Telematico | Esprime la partecipazione di un Dipendente ad un Meeting Telematico. | **Dipendente** [1..\*] ruolo partecipante : indica un Dipendente che partecipa ad un certo Meeting Telematico.  **Meeting Telematico** [0..\*] ruolo partecipa a : indica un Meeting Telematico a cui partecipa un determinato Dipendente. |
| Argomento Meeting Fisico | Esprime la relazione tra un Meeting Fisico ed il Progetto di cui si è discusso in esso. | **Progetto** [1] ruolo relativo a : indica il Progetto per cui si è tenuto un determinato Meeting Fisico.  **Meeting Fisico** [0..\*] ruolo discusso in : indica un Meeting Fisico in cui si è discusso di un determinato Progetto. |
| Argomento Meeting Telematico | Esprime la relazione tra un Meeting Telematico ed il Progetto di cui si è discusso in esso. | **Progetto** [1] ruolo relativo a : indica il Progetto per cui si è tenuto un determinato Meeting Telematico.  **Meeting Telematico** [0..\*] ruolo discusso in : indica un Meeting Telematico in cui si è discusso di un determinato Progetto. |
| Ambiti Progetto | Esprime la relazione tra un Progetto e i suoi Ambiti | **Progetto** [1] ruolo di : indica il Progetto a cui associamo un insieme di Ambiti.  **Ambito** [0..\*] ruolo relativo a : indica un Ambito associato ad un determinato Progetto. |
| Sala Prenotata | Esprime la relazione tra un Meeting Fisico e la Sala riunioni in cui esso si è svolto. | **Meeting Fisico** [0..\*] ruolo ospita : indica un Meeting Fisico che si è svolto in una determinata Sala riunioni.  **Sala** [1] ruolo svolto in : indica la Sala riunioni in cui un determinato Meeting Fisico si è svolto. |

## Dizionario dei Vincoli

|  |  |
| --- | --- |
| Nome | Descrizione |
| Codice Fiscale Legale | Il Codice Fiscale di un Dipendente deve rispettare la sua formattazione standard, ovvero deve contenere una sequenza di: 6 caratteri, 2 interi, 1 char, 2 int, 1 char, 3 int, 1 char. |
| Nome Legale | Il Nome di un Dipendente deve essere composto unicamente di caratteri. |
| Cognome Legale | Il Cognome di un Dipendente deve essere composto unicamente di caratteri. |
| Consistenza Valutazione | La valutazione aziendale di un Dipendente deve essere sempre uguale alla somma dei Progetti e dei Meeting a cui ha partecipato. |
| Consistenza Partecipante | Ciascun Partecipante ad un Progetto può avere al più 1 ruolo, ovvero lo stesso Dipendente non può partecipare ad uno stesso Progetto con più ruoli differenti. |
| Unicità Project Manager | Ad un Progetto può partecipare un solo Project Manager. |
| Totalità Project Manager | Ad un Progetto deve sempre partecipare un Project Manager. |
| Consistenza Meeting Fisico | Due Meeting Fisici distinti non si possono tenere nella stessa Sala, alla stessa ora dello stesso giorno. |
| Consistenza Partecipazione Meeting | Un Dipendente non può partecipare a più Meeting (Fisici/Telematici) contemporaneamente. |
| Capienza Sala Riunioni | Il numero di Partecipanti ad un Meeting Fisico non può eccedere il limite di posti della Sala in cui esso si svolge. |
| Limite Piattaforma Videoconferenze | Il numero di Partecipanti ad un Meeting Telematico non può eccedere il limite di partecipanti al Meeting (se esiste). |

# Progettazione Logica

In questo capitolo tratteremo la seconda fase della progettazione, scendendo ad un livello di astrazione più basso rispetto al precedente.

Lo schema concettuale verrà tradotto, anche grazie alla predisposizione conseguente la ristrutturazione, in uno schema logico, questa volta dipendente dalla struttura dei dati prescelta, nello specifico quella relazionale pura.

## Schema Logico

Di seguito è riportato lo schema logico della base di dati. Al suo interno, le chiavi primarie sono indicate con una sottolineatura singola mentre le chiavi esterne con una sottolineatura doppia.

* Dipendente (CodF, Nome, Cognome, Salario, Valutazione)
* Progetto (CodP, Tipologia)
* Ambito (CodP, Keyword)

CodP 🡪 Progetto.CodP

* Partecipante (CodF, CodP, Ruolo)

CodF 🡪Dipendente.CodF

CodP 🡪 Progetto.CodP

* Sala (CodSala, Città, Provincia, Indirizzo, NumCivico, NumPosti)
* MeetingF (CodMF, Data, OraI, OraF, CodSala)

CodSala 🡪 Sala.CodSala

* MeetingT (CodMT, Data, OraI, OraF, Piattaforma, NumLimite)
* PartecipaMF (CodF, CodMF)

CodF 🡪 Dipendente.CodF

CodMF 🡪 MeetingF.CodMF

* PartecipaMT (CodF, CodMT)

CodF 🡪 Dipendente.CodF

CodMT 🡪 MeetingT.CodMT

# Progettazione Fisica

In questo capitolo verrà riportata l’implementazione dello schema logico sopra descritto nel DBMS PostgreSQL.

## Definizione Tabelle

Di seguito sono riportate le definizioni delle tabelle e dei loro vincoli intrarelazionali.

### Definizione della Tabella DIPENDENTE

--Definizione tabella

CREATE TABLE DIPENDENTE

(

CodF CHAR(16) PRIMARY KEY DEFAULT 'AAAAAA00A00A000A'

CHECK (CodF ~\* '^[A-Za-z]{6}[0-9]{2}[A-Za-z][0-9]{2}[A-Za-z][0-9]{3}[A-Za-z]$'),

Nome VARCHAR(30) NOT NULL

CHECK (Nome ~\* '^[A-Za-z ]+$'),

Cognome VARCHAR(30) NOT NULL

CHECK (Cognome ~\* '^[A-Za-z ]+$'),

Salario REAL,

Valutazione INTEGER NOT NULL

);

### Definizione della Tabella PROGETTO

--Definizione tabella

CREATE TABLE PROGETTO

(

CodP INTEGER PRIMARY KEY,

Tipologia VARCHAR(30) NOT NULL,

Descrizione VARCHAR(200)

);

--Trigger per impostare, se necessario, la chiave primaria surrogata in automatico

CREATE OR REPLACE FUNCTION Funzione\_Sequenza\_Progetto()

RETURNS TRIGGER AS

$$

BEGIN

IF (NEW.CodP = 0) THEN

NEW.CodP := NEXTVAL('N\_PROGETTO');

END IF;

RETURN NEW;

END;

$$

LANGUAGE plpgsql;

CREATE TRIGGER Trigger\_Sequenza\_Progetto

BEFORE INSERT ON PROGETTO

FOR EACH ROW

EXECUTE PROCEDURE Funzione\_Sequenza\_Progetto();

### Definizione della Tabella AMBITO

CREATE TABLE AMBITO

(

CodP INTEGER ,

Keyword VARCHAR(30) NOT NULL,

CONSTRAINT fk\_ambito FOREIGN KEY (CodP)

REFERENCES PROGETTO(CodP)

ON UPDATE CASCADE

ON DELETE NO ACTION

);

### Definizione della Tabella PARTECIPANTE

CREATE TABLE PARTECIPANTE

(

CodF CHAR(16),

CodP INTEGER,

Ruolo VARCHAR(30) NOT NULL,

CONSTRAINT fk\_CodF FOREIGN KEY(CodF)

REFERENCES DIPENDENTE(CodF)

ON UPDATE CASCADE

ON DELETE NO ACTION,

CONSTRAINT fk\_CodP FOREIGN KEY(CodP)

REFERENCES PROGETTO(CodP)

ON UPDATE CASCADE

ON DELETE NO ACTION,

CONSTRAINT u\_Ruolo UNIQUE (CodF,CodP)

);

### Definizione della Tabella SALA

CREATE TABLE SALA

(

CodSala INTEGER PRIMARY KEY,

Città VARCHAR(30) NOT NULL CHECK (Città ~\* '^[A-Za-z ]+$'),

Provincia CHAR(2) NOT NULL CHECK (Provincia ~\* '^[A-Z]{2}$'),

Indirizzo VARCHAR(50) NOT NULL CHECK (Indirizzo ~\* '^[A-Za-z ]+$'),

NumCivico INTEGER,

NumPosti INTEGER NOT NULL

);

### Definizione della Tabella MEETINGF

CREATE TABLE MEETINGF

(

CodMF INTEGER PRIMARY KEY,

CodP INTEGER,

Data DATE NOT NULL,

OraI TIME NOT NULL,

OraF TIME NOT NULL,

CodSala INTEGER,

CONSTRAINT fk\_mf\_progetto FOREIGN KEY (CodP)

REFERENCES PROGETTO(CodP)

ON UPDATE CASCADE

ON DELETE NO ACTION,

CONSTRAINT fk\_sala FOREIGN KEY (CodSala)

REFERENCES SALA(CodSala)

ON UPDATE CASCADE

ON DELETE NO ACTION

);

### Definizione della Tabella MEETINGT

CREATE TABLE MEETINGT

(

CodMT INTEGER PRIMARY KEY,

CodP INTEGER,

Data DATE NOT NULL,

OraI TIME NOT NULL,

OraF TIME NOT NULL,

Piattaforma VARCHAR(30) NOT NULL,

NumLimite INTEGER,

CONSTRAINT fk\_mt\_progetto FOREIGN KEY (CodP)

REFERENCES PROGETTO(CodP)

ON UPDATE CASCADE

ON DELETE NO ACTION

);

### Definizione della Tabella PARTECIPAMF

CREATE TABLE PARTECIPAMF

(

CodF CHAR(16),

CodMF INTEGER,

CONSTRAINT u\_pmf UNIQUE(CodF, CodMF),

CONSTRAINT fk1\_MF FOREIGN KEY(CodF)

REFERENCES DIPENDENTE(CodF)

ON UPDATE CASCADE

ON DELETE NO ACTION,

CONSTRAINT fk2\_MF FOREIGN KEY(CodMF)

REFERENCES MEETINGF(CodMF)

ON UPDATE CASCADE

ON DELETE NO ACTION

);

### Definizione della Tabella PARTECIPAMT

CREATE TABLE PARTECIPAMT

(

CodF CHAR(16),

CodMT INTEGER,

CONSTRAINT u\_pmf UNIQUE(CodF, CodMT),

CONSTRAINT fk1\_MT FOREIGN KEY(CodF)

REFERENCES DIPENDENTE(CodF)

ON UPDATE CASCADE

ON DELETE NO ACTION,

CONSTRAINT fk2\_MT FOREIGN KEY(CodMT)

REFERENCES MEETINGT(CodMT)

ON UPDATE CASCADE

ON DELETE NO ACTION

);