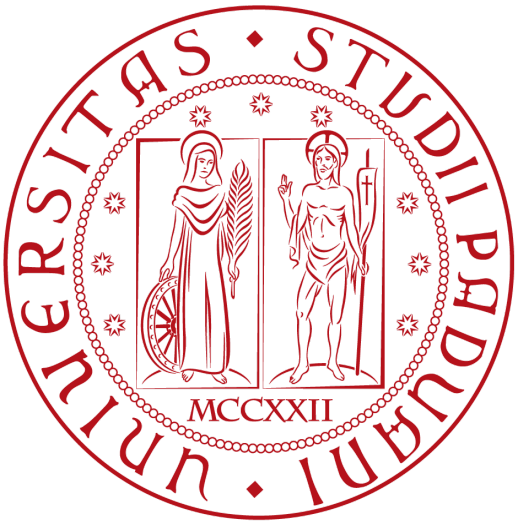


RELAZIONE BASI DI DATI

math@unipd

Cristian Maschio (1123398)
Andrea Favero (1125545)

DIPARTIMENTO
MATEMATICA

Anno Accademico 2016/2017

Indice

1- ABSTRACT

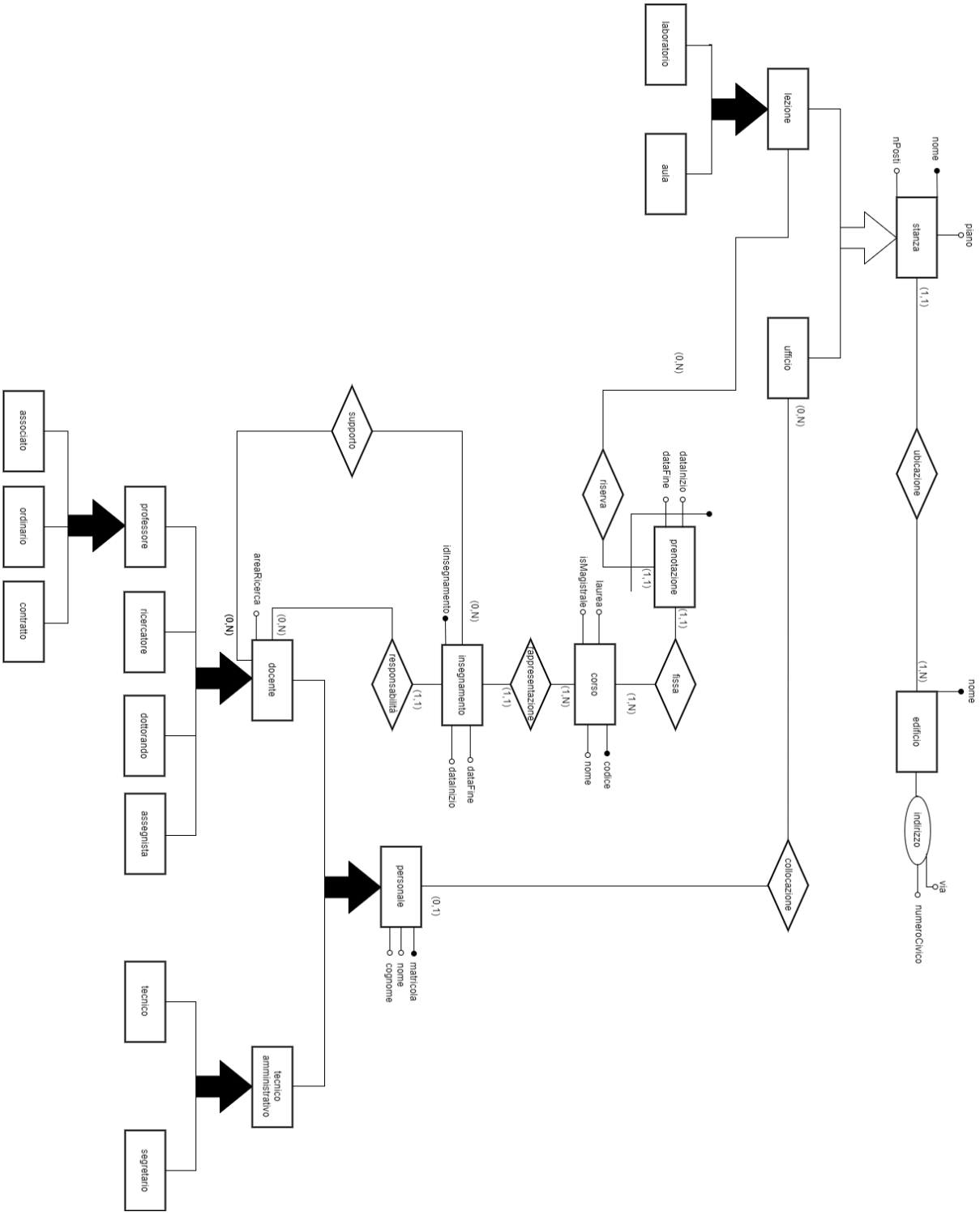
Il Dipartimento di Matematica Tullio Levi Civita è una struttura dell'Università degli Studi di Padova che
promuove e coordina le attività di ricerca nella Matematica e nell'Informatica. Esso è responsabile dei corsi di studio di Matematica ed Informatica (sia a livello di Laurea Triennale che Magistrale).
Al dipartimento afferisce un vasto numero di personale che si ripartisce in docenti, docenti esterni, assegnisti, dottorandi, personale tecnico amministrativo.
La sede del dipartimento è la Torre Archimede che si trova in via Trieste 63. Il dipartimento possiede anche delle aule e due laboratori informatici presso il Plesso Paolotti in via Belzoni 7 e via Luzzati 11.
math@unipd è una base di dati in grado di fornire informazioni sul personale che lavora nel dipartimento.
Permette di conoscere l'ubicazione del personale all'interno delle strutture del dipartimento e, i corsi che i docenti del dipartimento insegnano nelle lauree in Informatica e Matematica (sia triennali che magistrali) con la relativa prenotazione delle aule (se i corsi sono tenuti in aule del dipartimento).

2- ANALISI DEI REQUISITI

Il personale che opera nel Dipartimento di Matematica è composto da docenti che possono essere di una certa fascia (professori ordinari o professori associati o a contratto) oppure ricercatori , vi sono poi i dottorandi che possono fare da supporto alla didattica. Infine c'è il personale tecnico/amministrativo composto principalmente dai tecnici e dai/dalle segretari/arie. Del personale afferente al dipartimento ci interessano il nome ed anche il cognome, se si tratta di un docente allora vogliamo sapere anche la sua area di ricerca (ad esempio INF/01 per gli informatici ed MAT/03 per i matematici che fanno ricerca nel campo della geometria).
Poiché il dipartimento è responsabile per le lauree in Matematica ed Informatica esso offre dei corsi per gli studenti che le frequentano. Di ogni corso vogliamo sapere il nome (ad esempio Analisi 1, Analisi 2, Programmazione, Reti di Calcolatori, Ingegneria del Software, ecc..), se è della laurea in Matematica o in Informatica ed anche se è un corso per la triennale o per la magistrale.
I docenti (ma non i dottorandi) nel corso della loro carriera possono essere responsabili di alcuni insegnamenti; di ogni insegnamento ci interessano la data di inizio e quella di fine. I dottorandi

possono fare da supporto alla didattica per alcuni insegnamenti. Un certo insegnamento può essere associato ad un unico corso, ma un corso può essere associato a più insegnamenti. Ad esempio il Prof. Don Knuth può insegnare dal 15 Settembre al
21 Dicembre del 2017 il corso ”The Art of Computer Programming” per la laurea in Informatica ma, tale corso può essere tenuto nello stesso periodo anche dal Prof. Edsger Dijkstra, magari perchè ci sono troppi studenti ed allora si è scelto di fare più canali.
Per ogni corso si può fissare una prenotazione (per un certo periodo di tempo, ci interessano l'inizio e la fine) che andrà a riservare un'aula o un laboratorio del dipartimento. Le aule ed i laboratori sono delle stanze usate per fare lezione ma ne esistono anche altri tipi. Ad esempio gli uffici sono delle stanze in cui possono trovare collocazione gli afferenti del dipartimento. Un membro del personale può avere al massimo un ufficio (ma non è detto che ne abbia uno, ad esempio l'università potrebbe decidere di non assegnarlo ad un professore a contratto) e questo può essere condiviso con altre persone.
Ogni stanza è ubicata in un piano di un edificio del dipartimento (di cui si conoscono il nome e l'indirizzo), ed è identificata anche da un nome e dal numero di posti.

SCHEMA ER INIZIALE



3- DIZIONARIO DEI DATI

ENTITÀ	DESCRIZIONE	ATTRIBUTI	TIPO
edificio	Rappresenta un edifico che è del Dipartimento di Matematica	**Nome**	VARCHAR(16)
		Via	VARCHAR(30)
		NumeroCivico	VARCHAR(15)
stanza	Identifica una stanza ubicata in un edificio	Nome	VARCHAR(10)
		Piano	Int(11)
		NumeroPosti	Int(11)
lezione	Sono una generalizzazione di stanza. Le aule da lezione possono essere riservate per i corsi, l'ufficio invece puó essere occupato dal personale che lavora nel dipartimento	-	-
Ufficio			
personale	Rappresenta un afferente al dipartimento, puó essere un docente o un membro del personale tecnico amministrativo	**Matricola**	INT(7)
		Nome	VARCHAR(15)
		Cognome	VARCHAR(15)
tecnico_ amministrativo	Generalizzazione di personale	-	-
docente		Area	VARCHAR(20)
tecnico	Sono una generalizzazione di tecnico_amministrativo	-	- -
segretario			
assegnista	Sono una generalizzazione di docente	-	-
dottorando			
ricercatore			
professore			

associato	Sono una generalizzazione di professore	-	-
ordinario			
contratto			
corso	Modella un corso universitario per la laurea in Informatica o Matematica	**Codice**	INT(11)
		Nome	VARCHAR(30)
		Laurea	VARCHAR(30)
		isMagistrale	BOOLEAN
insegnamento	Rappresenta un corso in svolgimento in un determinato periodo, con un docente responsabile e dei docenti di supporto	**IdInsegnamento**	INT(7)
		DataInizio	DATE
		DataFine	DATE
prenotazione	Modella la prenotazione di un'aula da lezione da parte di un corso che intende utilizzarla	IdStanza	VARCHAR(10)
		IdCorso	INT(11)

RELAZIONE	DESCRIZIONE	ENTITÁ COINVOLTE	ATTRIBUTI	TIPO
ubicazione	associa una stanza con l'edificio edificio dell'universitá in cui è ubicata	stanza (1,1)		
		edificio (1, N)		
collocazione	associa il personale del dipartimento all'ufficio che gli è stato assegnato	ufficio (0, N)		
		personale (0,1)		
responsabilitá	associa un	docente (0,N)		

	docente ad uno degli insegnamenti che di cui é responsabile o di cui lo è stato	insegnamento (1,1)		
rappresentazione	associa ad un corso una prenotazione di un'aula	insegnamento (1, 1) corso (1, N)		
fissa	associa un'aula da lezione con una delle sue prenotazioni	corso (1,N) prenotazione (1,1)		
riserva	associa una prenotazione all'aula	prenotazione (1,1) lezione (0,N)		
supporto	associa un docente ad un insegnamento a cui fa da assistenza alla didattica	docente (0,N) insegnamento(0,N)	descrizione	VARCHAR (300)

4-Eliminazione delle gerarchie

Nello schema sono presenti 2 gerarchie: una riguarda le aule ed un'altra riguarda il personale che lavora nel dipartimento.
aule: la gerarchia alla fine diventa un'unica entitá grazie all'attributo tipoAula che permette di distinguere le aule per fare lezione dagli uffici dei tecnici o dei docenti.
personale: la gerarchia é stata sostituita da due nuove entitá, **docente** e **tecnico_amministrativo**. docente rappresenta l'insieme di tutti i docenti che erano contenuti nella sottogerarchia docente. tecnico_amministrativo rappresenta l'insieme dei tecnici e dei segretari che erano contenuti nella sottogerarchia tecnicnico_amministrativo. Per distinguere tra di loro i docenti sono stati aggiunti all'entitá docente l'attributo **categoria** (che serve per dire se i docenti sono professori o ricercatori ...) e
l'attributo **fascia** che serve per dire, nel caso il docente sia un professore se, é ordinario, associato o a contratto.

5-Descrizione dei vincoli referenziali

Nel processo di traduzione allo schema ER sono stati introdotti dei vincoli di integrità referenziale:
1. Ad un docente e ad un tecnico non è possibile assegnargli una stanza che non sia un ufficio.
2. L'insegnamento gli attributi IdCorso e IdDocente non possono assumere valori che non siano uguali ad almeno un valore assunto della chiave primaria dalle entità Corso e Docente.
3. La prenotazione gli attributi IdCorso e IdStanza non possono assumere valori che non siano uguali ad almeno un valore assunto dalla chiave primaria dalle entità Corso e Stanza, su IdStanza, non è possibile prenotare stanze di tipo Ufficio.

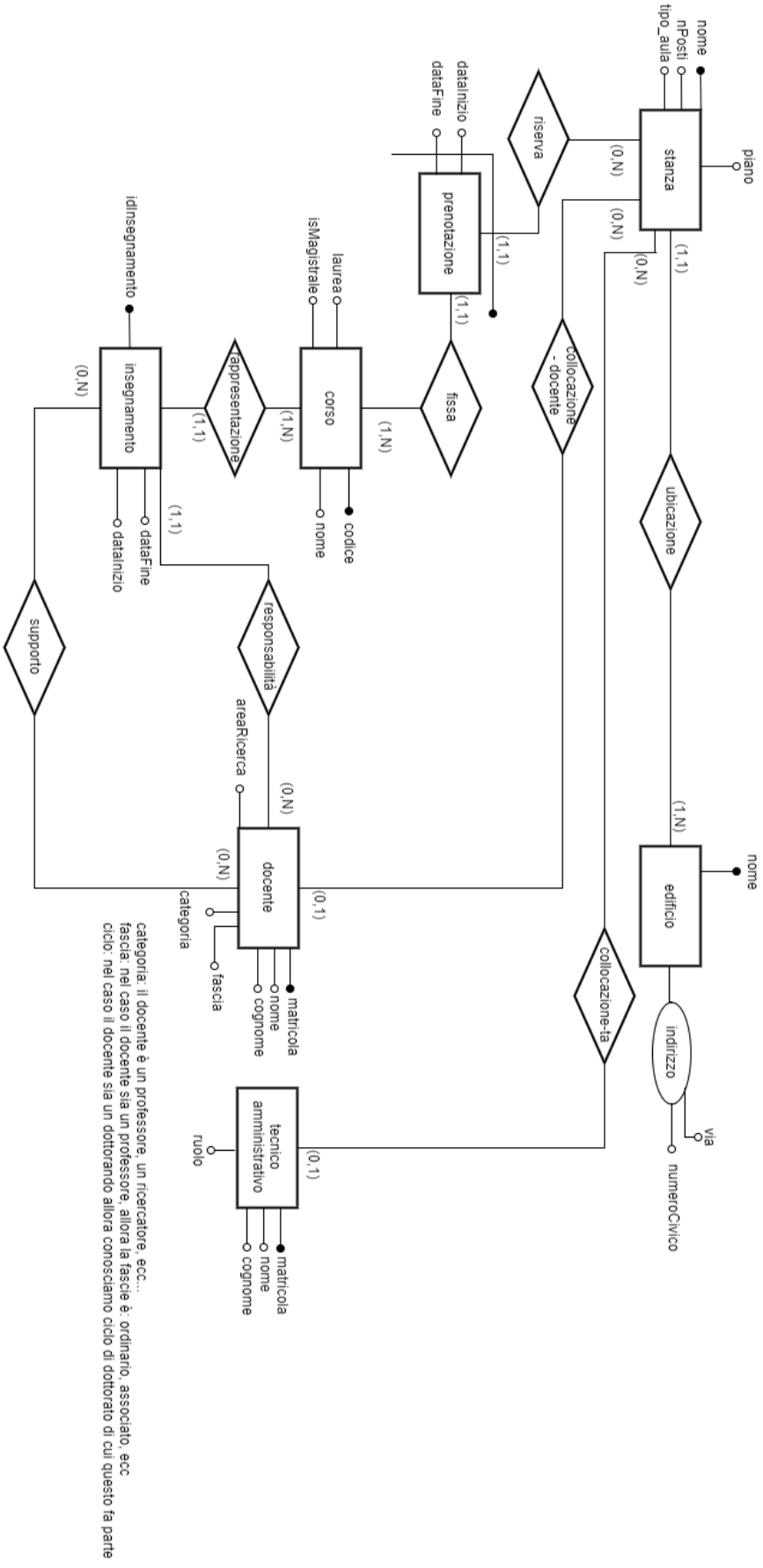
4.La stanza ha come attributo IdEdificio la quale non può assumere valori che non siano uguali ad almeno un valore assunto dalla chiave primaria dall'entità Edificio.
5.Il supporto ha come attributo IdInsegnamento e IdDocente le quali non possono assumere valori che non siano uguali ad almeno un valore assunto dalla chiave primaria dalle entità Insegnamento e Docente.

Traduzione dello schema E-R

Stanza(**Nome**, NumPosti, TipoStanza, Piano, IdEdificio)
Edificio(**Nome**, Via, NumeroCivico)
Docente(**Matricola**, Nome, Cognome, Categoria, Fascia, AreaRicerca, IdStanza*)
Insegnamento(**IdInsegnamento**, DataInizio, DataFine, IdCorso, IdDocente)
Prenotazione(**IdStanza**, IdCorso, **DataInizio**, **DataFine**)
Supporto(**IdDocenteSupporto**, **IdInsegnamento**, Descrizione*)
Tecnico_Amministrativo(**Matricola**, Nome, Cognome, Ruolo, IdStanza)

(*=attributi opzionali, di default NULL)

ER-RISTRUTTURATO



6-FUNZIONI

-Funzione la quale verifica se e' disponibile un'aula in un determinato lasso di tempo ritorna true se esiste già una prenotazione altrimenti false

```
DROP FUNCTION IF EXISTS CheckPrenotazione;
DELIMITER |
CREATE FUNCTION CheckPrenotazione(Stanza VARCHAR(10),Corso INT(11),DataInizio
DATETIME, DataFine DATETIME)
RETURNS BOOLEAN
BEGIN
        IF EXISTS (SELECT p.DataInizio
          FROM prenotazione AS p
          WHERE Stanza = p.IdStanza AND p.DataInizio<DataFine AND
p.DataFine>DataInizio)THEN
        RETURN 1;
   ELSE
        RETURN 0;
   END IF;
END|

DELIMITER ;
```

-Funzione la quale verifica se un ufficio ha un posto disponibile
Ritorna falso se ha posti disponibili
Ritorna true se non ha posti disponibili

```
DROP FUNCTION IF EXISTS CheckUffici;
DELIMITER |
CREATE FUNCTION CheckUffici(Ufficio VARCHAR(10))
RETURNS BOOLEAN
BEGIN
        IF (Ufficio IS NULL OR
        (SELECT COUNT(stanza.Nome) AS Numero
        FROM ((stanza LEFT JOIN docente ON stanza.Nome=docente.IdStanza) LEFT JOIN
tecnico_amministrativo ON stanza.Nome = tecnico_amministrativo.IdStanza)
        WHERE Ufficio = stanza.Nome)
        <
        (SELECT stanza.NumPosti
        FROM stanza
        WHERE Ufficio=stanza.Nome))THEN
        RETURN 0;
   ELSE
        RETURN 1;
   END IF;
END|
```

```
DELIMITER ;
```

7-TRIGGER

-Trigger il quale ha il compito di verificare:
se non gli sia stata assegnata al docente una stanza che corrisponda ad un'aula, laboratorio o sala riunioni
se la stanza assegnata ha raggiunto la capienza massima
se la Categoria corrisponde al professore, allora la fascia non può essere vuota
se la Categoria non corrisponde al professore, allora la fascia deve essere vuota

```
DROP TRIGGER IF EXISTS Before_Docente_Insert;
DELIMITER |
CREATE TRIGGER `Before_Docente_Insert` BEFORE INSERT ON `docente` FOR EACH ROW
BEGIN
DECLARE Tipo varchar(15);

DECLARE TipoC varchar(15);
SET TipoC = NEW.Categoria;

SELECT TipoStanza INTO Tipo
   FROM stanza
   WHERE Nome=NEW.IdStanza;

IF ('Aula' LIKE Tipo OR 'Laboratorio' LIKE Tipo OR 'Sala riunioni' LIKE Tipo) THEN
   SIGNAL SQLSTATE '45000'
   SET MESSAGE_TEXT = 'Impossibile aggiungere il docente, al docente e'' possibile assegnargli solo gli uffici';
END IF;
IF (CheckUffici(NEW.IdStanza)) THEN
        SIGNAL SQLSTATE '45000'
   SET MESSAGE_TEXT = 'Impossibile aggiungere il docente, l''ufficio ha gia'' raggiunto la capienza massima';
END IF;

IF ('Professore' LIKE TipoC) THEN
        IF ('' LIKE NEW.Fascia) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Impossibile aggiungere il docente, al professore va assegnata la fascia';
   END IF;
ELSE IF ('' NOT LIKE NEW.Fascia) THEN
        SIGNAL SQLSTATE '45000'
   SET MESSAGE_TEXT = 'Impossibile aggiungere il docente, la fascia va assegnata solo al professore';
END IF;
END IF;
```

```
END|
DELIMITER ;
```

-Ha il compito di verificare l'inserimento di una prenotazione corretta tramite il controllo:
Se l'aula selezionata e' corretta
se la data inzio e' < della data della fine prenotazione
se e' disponibile l'aula in quel lasso di tempo

```
DROP TRIGGER IF EXISTS Before_Prenotazione_Insert;
DELIMITER |
CREATE TRIGGER `Before_Prenotazione_Insert` BEFORE INSERT ON `prenotazione` FOR EACH
ROW BEGIN
DECLARE Tipo varchar(15);

SELECT TipoStanza INTO Tipo
   FROM stanza
   WHERE Nome=NEW.IdStanza;

IF ('Aula' LIKE Tipo OR 'Laboratorio' LIKE Tipo OR 'Sala riunioni' LIKE Tipo) THEN

   BEGIN
      IF(NEW.DataInizio >= NEW.DataFine) THEN
         SIGNAL SQLSTATE '45000'
         SET MESSAGE_TEXT = 'Impossibile aggiungere la prenotazione, la data inizio deve essere <
della data fine';
      END IF;

      IF (CheckPrenotazione(NEW.IdStanza, NEW.IdCorso, NEW.DataInizio, NEW.DataFine))THEN
         SIGNAL SQLSTATE '45000'
         SET MESSAGE_TEXT = 'Impossibile aggiungere la prenotazione, le date si sovrappongono';
      END IF;
          END;
ELSE
   SIGNAL SQLSTATE '45000'
   SET MESSAGE_TEXT = 'Impossibile aggiungere la prenotazione, gli uffici non possono essere
prenotati';
END IF;
END|
DELIMITER ;
```

-Ha il compito di verificare che al tecnico amministrativo non gli venga assegnata una stanza diversa dall'ufficio

```
DROP TRIGGER IF EXISTS Before_Tecnico_Insert;
DELIMITER |
```

```
CREATE TRIGGER `Before_Tecnico_Insert` BEFORE INSERT ON `tecnico_amministrativo` FOR
EACH ROW BEGIN
DECLARE Tipo varchar(15);

SELECT TipoStanza INTO Tipo
   FROM stanza
   WHERE Nome=NEW.IdStanza;

IF ('Aula' LIKE Tipo OR 'Laboratorio' LIKE Tipo OR 'Sala riunioni' LIKE Tipo) THEN
   SIGNAL SQLSTATE '45000'
   SET MESSAGE_TEXT = 'Impossibile aggiungere l''utente, al tecnico ammministrativo e'' possibile
assegnargli solo gli uffici';
END IF;

IF (CheckUffici(NEW.IdStanza)) THEN
        SIGNAL SQLSTATE '45000'
   SET MESSAGE_TEXT = 'Impossibile aggiungere il tecnico, l''ufficio ha gia'' raggiunto la capienza
massima';
END IF;

END|
DELIMITER ;
```

8-QUERY E PROCEDURE

-Mostra per ogni docente il corso di insegnamento
OUTPUT: Nome | Cognome | AreaRicerca | Categoria | Fascia | Ufficio | Corso

```
SELECT docente.Nome, docente.Cognome, docente.AreaRicerca, docente.Categoria,
docente.Fascia, docente.IdStanza AS Ufficio, corso.Nome AS Corso
FROM (docente LEFT JOIN insegnamento ON docente.Matricola=insegnamento.IdDocente) LEFT
JOIN corso ON insegnamento.IdCorso=corso.Codice;
```

-Mostra tutte le aule prenotate da ciascun corso in ordine di corso
OUTPUT: Corso | Aula | DataInizo | DataFine

```
SELECT corso.Nome AS Corso, prenotazione.IdStanza AS Aula, prenotazione.DataInizio AS
DataInzio, prenotazione.DataFine AS DataFine
FROM prenotazione LEFT JOIN corso ON prenotazione.IdCorso = corso.Codice
ORDER BY Corso
```

-Mostra tutte le aule prenotate in un certo lasso ti tempo
OUTPUT: Stanza | Corso | DataInizo | DataFine

```
SELECT IdStanza AS Stanza, corso.Nome AS Corso, DataInizio AS DataInizio, DataFine AS
DataFine
FROM prenotazione LEFT JOIN corso ON prenotazione.IdCorso = corso.Codice
WHERE (DataInizio >= '2017-10-20 00:00:00' AND DataInizio <= '2017-10-20 12:00:00') OR
      (DataFine >= '2017-10-20 00:00:00' AND DataFine <= '2017-10-20 12:00:00')
```

-Mostra tutte le aule libere in un certo lasso ti tempo
OUTPUT: Stanza

```
SELECT stanza.Nome AS Stanza
FROM stanza
WHERE Nome NOT IN(
   SELECT IdStanza AS Nome
   FROM prenotazione LEFT JOIN corso ON prenotazione.IdCorso = corso.Codice
   WHERE (DataInizio >= '2017-10-20 00:00:00' AND DataInizio <= '2017-10-20 12:00:00') OR
         (DataFine >= '2017-10-20 00:00:00' AND DataFine <= '2017-10-20 12:00:00'))
```

-Per ogni insegnamento, mostra, docente, supporto, con ulteriori informazioni su docente e supporto, in ordine di Corso
OUTPUT: Corso, DataInizio, DataFine, NomeDocente, CognomeDocente, CategoriaDocente, UfficioDocente, NomeSupporto, CognomeSupporto, CategoriaSupporto, UfficioSupporto

```
SELECT corso.Nome AS Corso, insegnamento.DataInizio, insegnamento.DataFine, d.Nome AS
NomeDocente,
      d.Cognome AS CognomeDocente, d.Categoria AS CategoriaDocente, d.IdStanza AS
UfficioDocente,
      docente.Nome AS NomeSupporto, docente.Cognome AS CognomeSupporto,
      docente.Categoria AS CategoriaSupporto, docente.IdStanza AS UfficioSupporto
FROM ((insegnamento LEFT JOIN supporto ON
insegnamento.IdInsegnamento=supporto.IdInsegnamento)
      LEFT JOIN docente ON supporto.IdDocenteSupporto=docente.Matricola)
      LEFT JOIN corso ON corso.Codice=insegnamento.IdCorso
      LEFT JOIN docente AS d ON d.Matricola=insegnamento.IdDocente
ORDER BY corso.Nome
```

-Inserisce piu' prenotazioni in un lasso di tempo, sempre nello stesso giorno della settimana, partendo da DataInizio con orari: OraInzio a Orafine fino al raggiungimento della DataFine

```
DROP PROCEDURE IF EXISTS PrenotazioneSettimanale;

DELIMITER |
CREATE PROCEDURE PrenotazioneSettimanale(Stanza VARCHAR(10),Corso INT(11),DataInizio
DATE, DataFine DATE, OraInizio TIME, OraFine TIME)

BEGIN
```

```
DECLARE DataProgress DATETIME;

        IF (DataInizio<DataFine AND OraInizio<OraFine) THEN
   SELECT DATE_ADD(DataInizio,INTERVAL 0 DAY) INTO DataProgress;
        WHILE (DataProgress<=DataFine) DO
        IF NOT(CheckPrenotazione(Stanza, Corso, DataProgress+OraInizio,
DataProgress+OraFine)) THEN
          INSERT INTO `prenotazione` (`IdStanza`, `IdCorso`, `DataInizio`, `DataFine`) VALUES
          (Stanza, Corso, DataProgress+OraInizio, DataProgress+OraFine);
        END IF;
        SELECT DATE_ADD(DataProgress,INTERVAL 7 DAY) INTO DataProgress;
                END WHILE;

   END IF;
END|
DELIMITER ;
```