Progetto d'Esame di Basi di Dati
Documentazione Tecnica

Simone Alessandro Casciaro
simonealex.casciaro@studenti.unimi.it

September 1, 2023

1 Tabelle

Spiegazione del significato delle colonne delle tabelle ed eventuali vincoli

1.1 Appello

nome colonna	tipo	descrizione	vincolo
data	timestamp	Indica la data e l'ora dell'appello	PRIMARY KEY
insegnamento	integer	L'ID dell'insegnamento	PRIMARY KEY e FOREIGN KEY Insegnamento("ID")
luogo	varchar	Il luogo in cui si svolge l'esame	

1.2 Corso

nome colonna	tipo	descrizione	vincolo
nome	varchar	Il nome del corso di laurea	PRIMARY KEY
durata	integer	2 = Corso Magistrale, 3 = Corso Triennale	
descrizione	text	Una breve descrizione sul corso	

1.3 Docente

nome colonna	tipo	vincolo	descrizione
username	varchar	La mail del docente	PRIMARY KEY
password	varchar	La password del docente, codificata in md5	
nome	varchar	Il nome del docente	
cognome	varchar	Il cognome del docente	
nascita	date	La data di nascita del docente	
sesso	varchar	Il sesso del docente	
indirizzo	varchar	L'indirizzo di casa del docente	

1.4 Esame

nome colonna	tipo	descrizione	vincolo
studente	integer	La matricola dello studente iscritto all'esame	PRIMARY KEY e FOREIGN KEY Studente("Matricola")
data	timestamp	La data e l'ora dell'appello	PRIMARY KEY e FOREIGN KEY Appello("Data")
insegnamento	integer	L'ID dell'insegnamento	PRIMARY KEY e FOREIGN KEY Appello("Insegnamento")
voto	integer	Il voto ottenuto dallo studente	
lode	boolean	Indica se lo studente ha ottenuto la lode	

1.5 Esame_Storico

nome colonna	tipo	descrizione	vincolo
studente	integer	La matricola dello studente iscritto all'esame	PRIMARY KEY e FOREIGN KEY Studente_Storico("Matricola")
data	timestamp	La data e l'ora dell'appello	PRIMARY KEY e FOREIGN KEY Appello("Data")
insegnamento	integer	L'ID dell'insegnamento	PRIMARY KEY e FOREIGN KEY Appello("Insegnamento")
voto	integer	Il voto ottenuto dallo studente	
lode	boolean	Indica se lo studente ha ottenuto la lode	

1.6 Insegnamento

nome colonna	tipo	descrizione	vincolo
ID	integer	Codice univoco dell'insegnamento	PRIMARY KEY
nome	varchar	Il nome dell'insegnamento	
anno	integer	L'anno in cui è prevista la fruizione	
descrizione	text	Una breve descrizione sull'insegnamento	
CFU	integer	Il numero di CFU che l'insegnamento fornisce	
corso	varchar	Il corso di cui l'insegnamento fa parte	FOREIGN KEY Corso("Nome")
responsabile	varchar	la mail del docente responsabile dell'insegnamento	FOREIGN KEY Insegnamento("Username")

1.7 Propedeuticita

nome colonna	tipo	descrizione	vincolo
insegnamento	integer	L'ID dell'insegnamento che richiede una propedeuticità	PRIMARY KEY e FOREIGN KEY Insegnamento("ID")
requisito	integer	L'ID dell'insegnamento propedeutico	PRIMARY KEY e FOREIGN KEY Insegnamento("ID")

1.8 Segreteria

nome colonna	tipo	vincolo	descrizione
username	varchar	La mail del segretario	PRIMARY KEY
password	varchar	La password del segretario, codificata in md5	
nome	varchar	Il nome del segretario	
cognome	varchar	Il cognome del segretario	
nascita	date	La data di nascita del segretario	
sesso	varchar	Il sesso del segretario	
indirizzo	varchar	L'indirizzo di casa del segretario	

1.9 Studente

nome colonna	tipo	vincolo	descrizione
matricola	integer	La matricola dello studente	PRIMARY KEY
username	varchar	La mail dello studente	UNIQUE
password	varchar	La password dello studente, codificata in md5	
nome	varchar	Il nome dello studente	
cognome	varchar	Il cognome dello studente	
nascita	date	La data di nascita dello studente	
sesso	varchar	Il sesso dello studente	
indirizzo	varchar	L'indirizzo di casa dello studente	
iscrizione	date	La data in cui lo studente si è iscritto	
corso	varchar	Il corso a cui è iscritto lo studente	FOREIGN KEY Corso("Nome")

1.10 Studente_Storico

nome colonna	tipo	vincolo	descrizione
matricola	integer	La matricola dello studente	PRIMARY KEY
username	varchar	La mail dello studente	UNIQUE
password	varchar	La password dello studente, codificata in md5	
nome	varchar	Il nome dello studente	
cognome	varchar	Il cognome dello studente	
nascita	date	La data di nascita dello studente	
sesso	varchar	Il sesso dello studente	
indirizzo	varchar	L'indirizzo di casa dello studente	
iscrizione	date	La data in cui lo studente si è iscritto	
inattivita	date	La data in cui lo studente si è disiscritto dall'università	
motivazione	varchar	La motivazione per la disiscrizione dall'università	
corso	varchar	Il corso a cui era iscritto lo studente	FOREIGN KEY Corso("Nome")

2 Funzioni PostgreSQL

2.1 add_docente

2.1.1 Descrizione

La funzione, attraverso i dati passati come argomenti, inserisce un nuovo record alla tabella "Docente" nel database mantenendolo consistente rispetto al requisito di avere almeno un insegnamento per ogni docente.
La funzione restituisce un valore boolean: TRUE se il docente viene inserito correttamente. FALSE altrimenti.
I casi in cui la funzione restituisce FALSE sono:
- Inserimento fallito per motivi riguardanti il dominio delle colonne del Docente
- Assegnare al nuovo docente un insegnamento già assegnato ad altri

2.1.2 Argomenti

- username varchar
- password varchar
- nome varchar

- cognome varchar
- data_di_nascita date
- sesso varchar
- indirizzo varchar
- primo_insegnamento integer

2.1.3 Output

- boolean

2.1.4 Codice

```
CREATE or REPLACE FUNCTION
add_docente(username varchar(200), password varchar(16), nome varchar(100),
cognome varchar(50), data_di_nascita date, sesso varchar(10),
indirizzo varchar(50), primo_insegnamento integer)
RETURNS BOOLEAN AS $$
DECLARE
    old_count integer;
    new_count integer;
BEGIN
    SELECT count(*) into old_count
    FROM docente;

    if (check_responsabile(primo_insegnamento) != 1) then
        return FALSE;
    END if;

    INSERT INTO docente VALUES
        (username, password, nome, cognome, data_di_nascita, sesso, indirizzo);

    UPDATE insegnamento
    SET responsabile = username
    WHERE id = primo_insegnamento;

    SELECT count(*) into new_count
    FROM docente;

    if (old_count=new_count) then
        ROLLBACK;
        return FALSE;
    else
        return TRUE;
    END if;

END;
$$ language 'plpgsql';
```

2.2 check_delete_docente

2.2.1 Descrizione

La funzione viene eseguita nel momento in cui si cerca di eliminare un docente. La funzione non impedisce di eliminare il docente, ma aggiorna gli insegnamenti associati togliendo il responsabile. Per farlo, disabilita momentaneamente il trigger relativo al numero di insegnamenti che un docente deve avere.

2.2.2 Argomenti

2.2.3 Output

- trigger

2.2.4 Codice

```
CREATE or REPLACE FUNCTION check_delete_docente() RETURNS TRIGGER AS $$
BEGIN

    ALTER TABLE insegnamento
    DISABLE check_numero_insegnamenti_trigger;

    UPDATE insegnamento
    SET responsabile = NULL
    WHERE responsabile = OLD.username;

    ALTER TABLE insegnamento
    ENABLE check_numero_insegnamenti_trigger;

    RETURN OLD;
END;
$$ language 'plpgsql';
```

2.3 check_delete_insegnamento

2.3.1 Descrizione

La funzione, prima di eliminare un insegnamento, controlla che non siano presenti studenti iscritti ad appelli relativi a quell'insegnamento. Se esistono, blocca l'eliminazione.

2.3.2 Argomenti

2.3.3 Output

- trigger

2.3.4 Codice

```
CREATE or REPLACE FUNCTION check_delete_insegnamento() RETURNS TRIGGER AS $$
DECLARE
    cnt integer;
BEGIN
```

```
  if EXISTS(
    SELECT *
    FROM esame
    WHERE insegnamento = OLD.id
  ) OR EXISTS (
    SELECT *
    FROM esame_storico
    WHERE insegnamento = OLD.id
  ) then raise exception 'Non puoi eliminare questo insegnamento';
  end if;

    SELECT count(*) into cnt
    FROM insegnamento
    WHERE responsabile = OLD.responsabile;

    if (cnt <= 1)
        then raise exception 'Il docente rimarrebbe privo di insegnamenti';
    RETURN OLD;
END;
$$ language 'plpgsql';
```

2.4 check_disiscrizione_esame

2.4.1 Descrizione

La funzione controlla che non sia possibile disiscriversi (e quindi eliminare un record dalla tabella Esame) da un appello passato.

2.4.2 Argomenti

2.4.3 Output

- trigger

2.4.4 Codice

```
CREATE or REPLACE FUNCTION check_disiscrizione_esame() RETURNS TRIGGER AS $$
BEGIN

  if (OLD.data::date <= NOW()::date)
    then raise exception 'Non ti puoi disiscrivere da questo esame';
  end if;

  RETURN OLD;
END;
$$ language 'plpgsql';
```

2.5 check_durata_corso

2.5.1 Descrizione

La funzione controlla che un'eventuale modifica alla durata del corso (che deve sempre rispettare il vincolo di essere 2 o 3) rimanga coerente con gli anni consigliati dagli insegnamenti che ne fanno parte.
Es. Non si può far diventare un corso da Triennale a Magistrale se in quel corso esiste un insegnamento che è previsto per il terzo anno.

2.5.2 Argomenti

2.5.3 Output

- trigger

2.5.4 Codice

```
CREATE or REPLACE FUNCTION check_durata_corso() RETURNS TRIGGER AS $$
BEGIN

    if EXISTS(
  SELECT *
  FROM insegnamento
  WHERE corso = OLD.nome AND anno > NEW.durata
    ) then
        raise exception 'Ci sono insegnamenti appartenenti al corso non conformi alla
                         modifica!';
    end if;

    RETURN NEW;
END;
$$ language 'plpgsql';
```

2.6 check_durata_insegnamento

2.6.1 Descrizione

La funzione, tramite un trigger, controlla che l'anno previsto dall'insegnamento sia coerente con la durata del corso associato.

2.6.2 Argomenti

2.6.3 Output

- trigger

2.6.4 Codice

```
CREATE or REPLACE FUNCTION check_durata_insegnamento() RETURNS TRIGGER AS $$
DECLARE
  anno_max integer;
BEGIN
```

```
  SELECT durata into anno_max
  FROM corso
  WHERE nome = NEW.corso;

  if (NEW.anno < 1 OR NEW.anno > anno_max) then
    raise exception 'Anno previsto non conforme';
  end if;

    if (OLD.corso <> NEW.corso) then
        raise exception 'Non si pu  modificare il corso di un insegnamento';
    end if;

  return NEW;
END;
$$ language 'plpgsql';
```

2.7 check_inserimento_appello

2.7.1 Descrizione

La funzione, tramite un trigger, impedisce l'inserimento di appelli con data passata.

2.7.2 Argomenti

2.7.3 Output

- trigger

2.7.4 Codice

```
CREATE or REPLACE FUNCTION check_inserimento_appello() RETURNS TRIGGER AS $$
BEGIN

  if (NEW.data::date < NOW()::date)
    then raise exception 'Impossibile inserire un appello precedente a oggi';
  end if;

  RETURN NEW;
END;
$$ language 'plpgsql';
```

2.8 check_inserimento_studente

2.8.1 Descrizione

La funzione, tramite un trigger, controlla che un nuovo studente inserito nel database non abbia la matricola o l'username uguale a quella di un altro studente nella tabella Studente_Storico e in caso ne impedisce l'inserimento.

2.8.2 Argomenti

2.8.3 Output

- trigger

2.8.4 Codice

```
CREATE or REPLACE FUNCTION check_inserimento_studente() RETURNS TRIGGER AS $$
BEGIN

    if EXISTS(
        SELECT *
        FROM studente_storico
        WHERE matricola = NEW.matricola OR username = NEW.username
    ) then raise exception 'Alcuni dati sono gi  stati usati da uno studente in passato';
    end if;
    if (NEW.corso <> OLD.corso)
        then raise exception 'Non si pu  modificare il corso di uno studente';
  end if;

  RETURN NEW;

END;
$$ language 'plpgsql';
```

2.9 check_inserimento_voti

2.9.1 Descrizione

La funzione, tramite un trigger, impedisce l'inserimento di voti ad appelli con data futura.

2.9.2 Argomenti

2.9.3 Output

- trigger

2.9.4 Codice

```
CREATE or REPLACE FUNCTION check_inserimento_voti() RETURNS TRIGGER AS $$
BEGIN

  if (NEW.data::date > NOW()::date)
    then raise exception 'Non si pu  modificare il voto di un esame non ancora svolto';
  end if;

  RETURN NEW;
END;
$$ language 'plpgsql';
```

2.10 check_iscrizione_esami

2.10.1 Descrizione

La funzione, tramite un trigger, impedisce l'iscrizione di uno studente ad appelli con data passata. Inoltre, ne impedisce l'iscrizione qualora l'insegnamento non faccia parte degli esami del suo corso di laurea e nel caso in cui lo studente non abbia superato tutti gli esami propedeutici per quell'insegnamento.

2.10.2 Argomenti

2.10.3 Output

- trigger

2.10.4 Codice

```
1  CREATE or REPLACE FUNCTION check_iscrizione_esami() RETURNS TRIGGER AS $$
2  DECLARE
3    laurea varchar(50);
4  BEGIN
5    SELECT corso into laurea
6    FROM studente
7    WHERE matricola = NEW.studente;
8
9    if NOT EXISTS(
10     SELECT *
11     FROM insegnamento
12     WHERE id = NEW.insegnamento AND corso = laurea
13   ) then
14     raise exception 'Insegnamento non presente nel corso di laurea dello studente';
15   end if;
16
17   if NOT check_propedeuticita(NEW.insegnamento, NEW.studente) then
18     raise exception 'Propedeuticit  esame non rispettate';
19   end if;
20
21   if (NEW.data::date < NOW()::date)
22     then raise exception 'Non    possibile iscriversi a un esame gi   passato';
23   end if;
24
25   return NEW;
26
27 END;
28 $$ language 'plpgsql';
```

2.11 check_numero_insegnamenti

2.11.1 Descrizione

La funzione, tramite un trigger, controlla che il vincolo del numero di insegnamenti per un docente sia sempre rispettato anche in caso di modifiche o di inserimenti di nuovi insegnamenti. In particolare, la funzione impedisce l'aggiunta o la modifica se il docente responsabile ha meno di un insegnamento o più di tre.

2.11.2 Argomenti

2.11.3 Output

- trigger

2.11.4 Codice

```
CREATE or REPLACE FUNCTION check_numero_insegnamenti() RETURNS TRIGGER AS $$
DECLARE
  new_count integer;
  old_count integer;
BEGIN
  SELECT count(*) into new_count
  FROM insegnamento
  WHERE responsabile = NEW.responsabile;

  if (TG_OP = 'INSERT') then
    if (new_count>=3) then
      raise exception 'Il docente ha gi  3 insegnamenti associati';
    end if;
  elsif (TG_OP = 'UPDATE') then
    SELECT count(*) into old_count
    FROM insegnamento
    WHERE responsabile = OLD.responsabile;

    if (old_count <= 1) then
      raise exception 'Il vecchio docente non avrebbe pi  insegnamenti';
    end if;
    if ((OLD.responsabile <> NEW.responsabile OR OLD.responsabile IS NULL) AND new_count >= 3) then
      raise exception 'Il nuovo docente ha gi  3 insegnamenti associati';
    end if;
  end if;

  return NEW;
END;
$$ language 'plpgsql';
```

2.12 check_propedeuticita

2.12.1 Descrizione

La funzione, prendendo come parametri l'id di un insegnamento e la matricola di uno studente, controlla se lo studente ha superato tutti gli esami per quell'insegnamento e restituisce un valore booleano.
TRUE se lo studente ha superato tutti gli esami propedeutici, FALSE altrimenti.

2.12.2 Argomenti

- codice_insegnamento integer
- matricola integer

2.12.3 Output

- boolean

2.12.4 Codice

```
CREATE or REPLACE FUNCTION
check_propedeuticita(codice_insegnamento integer, matricola integer)
RETURNS BOOLEAN AS $$
BEGIN

  if EXISTS(
    SELECT requisito
    FROM get_propedeuticita(codice_insegnamento)
    EXCEPT
    SELECT codice
    FROM get_carriera_valida(matricola)
  ) then
    return FALSE;
  else
    return TRUE;
    end if;

END;
$$ language 'plpgsql';
```

2.13 check_propedeuticita_corretta

2.13.1 Descrizione

La funzione, tramite un trigger, controlla che l'inserimento di una propedeuticità rispetti i seguenti requisiti:
- I due insegnamenti devono appartenere allo stesso corso
- Non ci possono essere catene di propedeuticià (ES. A è propedeutico per B, B è propedeutico per C, C è propedeutico per A)

2.13.2 Argomenti

2.13.3 Output

- trigger

2.13.4 Codice

```
CREATE or REPLACE FUNCTION check_propedeuticita_corretta() RETURNS TRIGGER AS $$
DECLARE
  corso_a varchar;
  corso_b varchar;
BEGIN
  SELECT corso into corso_a
  FROM insegnamento
  WHERE id = NEW.insegnamento;
```

```
9
10    SELECT corso into corso_b
11    FROM insegnamento
12    WHERE id = NEW.requisito;
13
14    if (corso_b <> corso_a) then
15      raise exception 'I due insegnamenti appartengono a corsi diversi!';
16    end if;
17
18    if EXISTS(
19      WITH RECURSIVE catena AS (
20        SELECT *
21        FROM propedeuticita
22        WHERE insegnamento = NEW.requisito
23        UNION
24        SELECT c.insegnamento, p.requisito
25        FROM propedeuticita p
26        INNER JOIN catena c ON c.requisito = p.insegnamento
27      )
28      SELECT *
29      FROM catena
30      WHERE NEW.insegnamento = requisito
31    ) then raise exception 'Impossibile introdurre una catena di propedeuticit ';
32
33    end if;
34
35    return NEW;
36
37  END;
38  $$ language 'plpgsql';
```

2.14 check_responsabile

2.14.1 Descrizione

La funzione, che prende come parametro l'id di un insegnamento, restituisce un valore intero che dipende dallo stato del suo responsabile o dell'insegnamento stesso.
- Ritorna 0 se l'insegnamento non esiste.
- Ritorna 1 se l'insegnamento esiste ma non ha un docente responsabile.
- Ritorna 2 se l'insegnamento esiste e ha un docente responsabile.

2.14.2 Argomenti

- codice integer

2.14.3 Output

- integer

2.14.4 Codice

```
CREATE or REPLACE FUNCTION check_responsabile(codice integer) RETURNS integer AS $$

DECLARE
    docente varchar;
BEGIN

    if NOT EXISTS(
      SELECT *
      FROM insegnamento
      WHERE id = codice
    ) then
        return 0;
    end if;

    SELECT responsabile into docente
    FROM insegnamento
    WHERE id = codice;

    if (docente IS NULL) then
        return 1;
    else
        return 2;
    END if;

END;
$$ language 'plpgsql';
```

2.15 check_sovrapposizione_appello

2.15.1 Descrizione

La funzione, tramite un trigger, controlla se l'inserimento di un nuovo appello sia conforme ai requisit posti dal tema d'esame. Un appello non è conforme se esiste un altro appello dello stesso anno e dello stesso corso di quello che si sta inserendo.

2.15.2 Argomenti

2.15.3 Output

- trigger

2.15.4 Codice

```
CREATE or REPLACE FUNCTION check_sovrapposizione_appello() RETURNS TRIGGER AS $$
DECLARE
  an integer;
  co varchar;
BEGIN

  SELECT anno, corso into an, co
  FROM insegnamento
  WHERE id = NEW.insegnamento;
```

```
10
11    if EXISTS(
12      SELECT * into chk
13      FROM appello a
14      INNER JOIN insegnamento i ON a.insegnamento = i.id
15      WHERE i.anno = an AND i.corso = co AND a.data::date = NEW.data::date
16                            AND NEW.data::date <> OLD.data::date
17    ) then
18      raise exception 'Insegnamento dello stesso corso e anno gi  presente in questa data';
19    end if;
20
21    RETURN NEW;
22
23  END;
24  $$ language 'plpgsql';
```

2.16 check_storici

2.16.1 Descrizione

La funzione, tramite un trigger, lancia un eccezione in quanto non è possibile modificare o eliminare i dati storici.

2.16.2 Argomenti

2.16.3 Output

- trigger

2.16.4 Codice

```
1  CREATE or REPLACE FUNCTION check_storici() RETURNS TRIGGER AS $$
2  BEGIN
3      raise exception 'Impossibile eliminare o modificare questi dati';
4  END;
5  $$ language 'plpgsql';
```

2.17 delete_studente

2.17.1 Descrizione

La funzione, che prende come parametri la matricola di uno studente, la data in cui lo studente dichiara la sua volontà di disiscriversi dall'università e la motivazione per cui si disiscrive, procede a eliminare dalla tabella Studente i suoi dati. Inoltre, questa funzione attiva il trigger "spostamento_dati_studente_trigger" e, di conseguenza, aggiorna i campi "inattivita" e "motivazione" della tabella Studente_Storico con i parametri passati a questa funzione.

2.17.2 Argomenti

- codice integer
- inat date
- mot varchar

2.17.3 Output

- boolean

2.17.4 Codice

```
CREATE or REPLACE FUNCTION delete_studente(codice integer, inat date, mot varchar)
RETURNS BOOLEAN AS $$
DECLARE
    old_count integer;
    new_count integer;
BEGIN
  SELECT count(*) into old_count
  FROM Studente;

    DELETE FROM studente
    WHERE matricola = codice;

    ALTER TABLE studente_storico
    DISABLE TRIGGER check_storici_trigger;

    UPDATE studente_storico
    SET inattivita = inat,
        motivazione = mot
    WHERE matricola = codice;

    ALTER TABLE studente_storico
    ENABLE TRIGGER check_storici_trigger;

    SELECT count(*) into new_count
    FROM studente;

    if (old_count=new_count) then
    return FALSE;
  else
    return TRUE;
    END if;

END;
$$ language 'plpgsql';
```

2.18 get_carriera

2.18.1 Descrizione

La funzione, presa come parametro la matricola di uno studente (sia esso attivo o "Storico"), restituisce una tabella con tutti gli appelli ai quali lo studente si è iscritto e, di questi, i seguenti dati:
- ID dell'insegnamento
- Nome dell'insegnamento
- Voto che lo studente ha ottenuto in quell'esame
- L'eventuale lode (TRUE se l'ha ricevuta, FALSE altrimenti)
- La data e l'ora in cui lo studente ha affrontato l'esame

2.18.2 Argomenti

- cod integer

2.18.3 Output

- codice integer
- nome varchar
- voto integer
- lode boolean
- data timestamp

2.18.4 Codice

```
CREATE or REPLACE FUNCTION get_carriera(cod integer)
RETURNS TABLE(codice integer, nome varchar(100), voto integer, lode boolean, data timestamp)
AS $$

BEGIN
  if EXISTS(
    SELECT *
    FROM studente
    WHERE matricola = cod
  ) then
    return QUERY
    SELECT i.id, i.nome, e.voto, e.lode, e.data
    FROM esame e
    INNER JOIN insegnamento i ON i.id = e.insegnamento
    WHERE e.studente = cod
    ORDER BY e.data;
  else if EXISTS(
    SELECT *
    FROM studente_storico
    WHERE matricola = cod
  ) then
    return QUERY
    SELECT i.id, i.nome, es.voto, es.lode, es.data
    FROM esame_storico es
    INNER JOIN insegnamento i ON i.id = es.insegnamento
    WHERE es.studente = cod
    ORDER BY es.data;
  else
    raise exception 'Studente inesistente';
  end if;
  end if;

  return;

END;
$$ language 'plpgsql';
```

2.19 get_carriera_valida

2.19.1 Descrizione

La funzione, presa come parametro la matricola di uno studente (sia esso attivo o "Storico"), restituisce una tabella con tutti gli appelli ai quali lo studente si è iscritto e che sono considerati "Validi". Per essere valido, un appello deve essere l'ultimo in ordine temporale per quell'insegnamento e deve comprendere un voto superiore o uguale a 18.
Degli appelli validi, sono forniti i seguenti dati:
- ID dell'insegnamento
- Nome dell'insegnamento
- Voto che lo studente ha ottenuto in quell'esame
- L'eventuale lode (TRUE se l'ha ricevuta, FALSE altrimenti)
- La data e l'ora in cui lo studente ha affrontato l'esame

2.19.2 Argomenti

- cod integer

2.19.3 Output

- codice integer
- nome varchar
- voto integer
- lode boolean
- data timestamp

2.19.4 Codice

```
CREATE or REPLACE FUNCTION get_carriera_valida(cod integer)
RETURNS TABLE(codice integer, nome varchar(100), voto integer, lode boolean, data timestamp)
AS $$
BEGIN

  if EXISTS(
    SELECT *
    FROM studente
    WHERE matricola = cod
  ) then
    return QUERY
    SELECT i.id, i.nome, e.voto, e.lode, e.data
    FROM esame e
    INNER JOIN insegnamento i ON i.id = e.insegnamento
    WHERE e.studente = cod
    AND e.data=(
      SELECT max(ee.data)
      FROM esame ee
      WHERE i.id = ee.insegnamento AND ee.studente = cod
```

```
    )
    AND e.voto >= 18
    ORDER BY e.data;
  else if EXISTS(
    SELECT *
    FROM studente_storico
    WHERE matricola = cod
  ) then
    return QUERY
    SELECT i.id, i.nome, es.voto, es.lode, es.data
    FROM esame_storico es
    INNER JOIN insegnamento i ON i.id = es.insegnamento
    WHERE es.studente = cod
    AND es.data=(
      SELECT max(ees.data)
      FROM esame_storico ees
      WHERE i.id = ees.insegnamento AND ees.studente = cod
    )
    AND es.voto >= 18
    ORDER BY es.data;
  else
    raise exception 'studente inesistente';
  end if;
  end if;

  return;

END;
$$ language 'plpgsql';
```

2.20 get_data

2.20.1 Descrizione

La funzione, presa come parametro la matricola di uno studente (sia esso attivo o "Storico"), restituisce una tabella con il numero di CFU ottenuti dallo studente (somma dei CFU degli insegnamenti presenti nella carriera valida dello studente) e la sua media (media dei voti ponderata dai CFU degli insegnamenti presenti nella carriera valida dello studente).

2.20.2 Argomenti

- student integer

2.20.3 Output

- CFU integer
- average float

2.20.4 Codice

```
CREATE or REPLACE FUNCTION get_data(student integer)
RETURNS TABLE(CFU bigint, average float) AS $$
BEGIN

  RETURN QUERY

  SELECT sum(i.CFU) as CFU, sum(c.voto*i.CFU)::float/sum(i.CFU) as average
  FROM get_carriera_valida(student) c
  INNER JOIN insegnamento i ON c.codice = i.id;

  RETURN;

END;
$$ language 'plpgsql';
```

2.21 get_info_corso

2.21.1 Descrizione

La funzione, preso come parametro il nome di un corso, restituisce tutte le informazioni relative agli insegnamenti che lo compongono, tra cui:
- L'id dell'insegnamento
- Il nome dell'insegnamento
- I CFU che l'insegnamento fornisce
- Il nome e il cognome del docente responsabile
- La descrizione dell'insegnamento

2.21.2 Argomenti

- nome_corso varchar

2.21.3 Output

- id integer
- nome varchar
- CFU integer
- responsabile text
- descrizione text

2.21.4 Codice

```
CREATE or REPLACE FUNCTION get_info_corso(nome_corso varchar)
RETURNS TABLE(id integer, nome varchar, CFU integer, responsabile text, descrizione text)
AS $$
BEGIN
  return QUERY

  SELECT i.id, i.nome, i.CFU, d.nome || ' ' || d.cognome AS doc, i.descrizione
```

```
  FROM insegnamento i
  LEFT JOIN docente d ON i.responsabile = d.username
  WHERE corso = nome_corso;

  return;

END;
$$ language 'plpgsql';
```

2.22 get_new_matricola

2.22.1 Descrizione

La funzione restituisce una nuova matricola valida da utilizzare per l'inserimento di un nuovo studente. Per farlo, calcola la matricola di valore maggiore e aggiunge 1, garantendo unicità.

2.22.2 Argomenti

2.22.3 Output

- integer

2.22.4 Codice

```
CREATE or REPLACE FUNCTION get_new_matricola() RETURNS integer AS $$
DECLARE
  output integer;
BEGIN

  WITH un AS (
    SELECT matricola
    FROM studente
    UNION
    SELECT matricola
    FROM studente_storico
  )
  SELECT max(matricola)+1 into output
  FROM un;

  RETURN output;

END;
$$ language 'plpgsql';
```

2.23 get_propedeuticita

2.23.1 Descrizione

La funzione, preso come parametro l'ID di un insegnamento, restituisce l'ID e il nome degli insegnamenti propedeutici per quell'insegnamento.

2.23.2 Argomenti

- codice_insegnamento integer

2.23.3 Output

- requisito integer
- nome varchar

2.23.4 Codice

```
CREATE or REPLACE FUNCTION get_propedeuticita(codice_insegnamento integer)
RETURNS TABLE(requisito integer, nome varchar(100)) AS $$
BEGIN
  return QUERY

  SELECT i.id, i.nome
  FROM insegnamento i
  INNER JOIN propedeuticita p ON p.requisito = i.id
  WHERE p.insegnamento = codice_insegnamento;

  return;

END;
$$ language 'plpgsql';
```

2.24 spostamento_dati_studente

2.24.1 Descrizione

La funzione, tramite un trigger, assicura che i dati dello studente che si vuole eliminare vengano invece spostati nelle rispettive tabelle "Storico". I dati generali dello studente vengono trasferiti nella tabella Studente_Storico, mentre tutta la sua carriera (ovvero gli appelli a cui si è iscritto e i relativi voti) vengono trasferiti nella tabella Esame_Storico. Per farlo, è necessario disabilitare momentaneamente il trigger check_disiscrizione_esame_trigger, in quanto altrimenti non sarebbe possibile eliminare dei record relativi ad appelli passati dalla tabella "Esame".

2.24.2 Argomenti

2.24.3 Output

- trigger

2.24.4 Codice

```
CREATE or REPLACE FUNCTION spostamento_dati_studente() RETURNS TRIGGER AS $$
BEGIN

  INSERT INTO studente_storico
  (
    SELECT *, NULL, NULL
```

```
    FROM studente
    WHERE matricola = OLD.matricola
  );

  INSERT INTO esame_storico
  (
    SELECT *
    FROM esame
    WHERE studente = OLD.matricola
  );

  ALTER TABLE esame
  DISABLE TRIGGER check_disiscrizione_esame_trigger;

  DELETE FROM esame
  WHERE studente = OLD.matricola;

  ALTER TABLE esame
  ENABLE TRIGGER check_disiscrizione_esame_trigger;

  return OLD;

END;
$$ language 'plpgsql';
```

3 Trigger

3.1 check_inserimento_appello_trigger

3.1.1 Tabella

Appello

3.1.2 Azione

BEFORE INSERT or UPDATE FOR EACH ROW

3.1.3 Funzione da eseguire

check_inserimento_appello

3.1.4 Descrizione

Questo trigger serve per impedire di aggiungere (o modificare) un appello in modo tale che la sua data risulti precedente al giorno in cui si effettua l'inserimento (o la modifica).

3.2 check_sovrapposizione_appello_trigger

3.2.1 Tabella

Appello

3.2.2 Azione

BEFORE INSERT or UPDATE FOR EACH ROW

3.2.3 Funzione da eseguire

check_sovrapposizione_appello

3.2.4 Descrizione

Questo trigger serve per impedire di aggiungere (o modificare) un appello in modo tale che esso risulti "sovrapposto" (quindi nella stessa data) a un esame dello stesso corso e dello stesso anno.

3.3 check_durata_corso_trigger

3.3.1 Tabella

Corso

3.3.2 Azione

BEFORE UPDATE FOR EACH ROW

3.3.3 Funzione da eseguire

check_durata_corso

3.3.4 Descrizione

Questo trigger serve per impedire modificare un corso in modo tale che la sua durata non sia coerente con gli anni previsti degli insegnamenti associati al corso.

3.4 check_delete_docente_trigger

3.4.1 Tabella

Docente

3.4.2 Azione

BEFORE DELETE FOR EACH ROW

3.4.3 Funzione da eseguire

check_delete_docente

3.4.4 Descrizione

Questo trigger serve per rendere NULL i campi "responsabile" di tutti gli insegnamenti che avevano il docente eliminato come responsabile.

3.5 check_disiscrizione_esame_trigger

3.5.1 Tabella

Esame

3.5.2 Azione

BEFORE DELETE FOR EACH ROW

3.5.3 Funzione da eseguire

check_disiscrizione_esame

3.5.4 Descrizione

Questo trigger serve per impedire a uno studente di disiscriversi da un esame passato.

3.6 check_inserimento_voti_trigger

3.6.1 Tabella

Esame

3.6.2 Azione

BEFORE UPDATE FOR EACH ROW

3.6.3 Funzione da eseguire

check_inserimento_voti

3.6.4 Descrizione

Questo trigger serve per impedire a un docente di inserire voti per un esame non ancora svolto.

3.7 check_iscrizione_esame_trigger

3.7.1 Tabella

Esame

3.7.2 Azione

BEFORE INSERT FOR EACH ROW

3.7.3 Funzione da eseguire

check_iscrizione_esame

3.7.4 Descrizione

Questo trigger serve per impedire a uno studente di iscriversi a un esame passato e a impedirgli di iscriversi se non rispetta le condizioni per farlo (Appartenere allo stesso corso e superare tutti gli esami propedeutici).

3.8 check_delete_insegnamento_trigger

3.8.1 Tabella

Insegnamento

3.8.2 Azione

BEFORE DELETE FOR EACH ROW

3.8.3 Funzione da eseguire

check_delete_insegnamento

3.8.4 Descrizione

Questo trigger serve per impedire l'eliminazione di un insegnamento qualora esistono studenti che risultano iscritti a un esame di quell'insegnamento.

3.9 check_durata_insegnamento_trigger

3.9.1 Tabella

Insegnamento

3.9.2 Azione

BEFORE INSERT or UPDATE FOR EACH ROW

3.9.3 Funzione da eseguire

check_durata_insegnamento

3.9.4 Descrizione

Questo trigger serve per impedire di impostare come "anno previsto" dell'insegnamento un anno non coerente con la durata del corso associato.

3.10 check_numero_insegnamenti_trigger

3.10.1 Tabella

Insegnamento

3.10.2 Azione

BEFORE INSERT or UPDATE FOR EACH ROW

3.10.3 Funzione da eseguire

check_numero_insegnamenti

3.10.4 Descrizione

Questo trigger serve per garantire che un docente abbia sempre almeno un insegnamento di cui essere responsabile e che abbia sempre al massimo tre insegnamenti di cui essere responsabile.

3.11 check_propedeuticita_corretta_trigger

3.11.1 Tabella

Propedeuticita

3.11.2 Azione

BEFORE INSERT or UPDATE FOR EACH ROW

3.11.3 Funzione da eseguire

check_propedeuticita_corretta

3.11.4 Descrizione

Questo trigger serve per impedire l'inserimento (o la modifica) di propedeuticità non valide. Una propedeuticità deve sempre riguardare due insegnamenti dello stesso corso e non devono esistere catene di propedeuticità.

3.12 check_inserimento_studente_trigger

3.12.1 Tabella

Studente

3.12.2 Azione

BEFORE INSERT or UPDATE FOR EACH ROW

3.12.3 Funzione da eseguire

check_inserimento_studente

3.12.4 Descrizione

Questo trigger serve per impedire l'inserimento (o la modifica) di uno studente i cui dati di riconoscimento (matricola e username) sono già stati utilizzati da un altro studente (attivo o "Storico").

3.13 spostamento_dati_studente_trigger

3.13.1 Tabella

Studente

3.13.2 Azione

BEFORE DELETE FOR EACH ROW

3.13.3 Funzione da eseguire

spostamento_dati_studente

3.13.4 Descrizione

Questo trigger serve per trasferire tutti i dati dello Studente da eliminare dentro le tabelle Studente_Storico ed Esame_Storico.

3.14 check_storici_trigger

3.14.1 Tabella

Studente_Storico, Esame_Storico

3.14.2 Azione

BEFORE DELETE or UPDATE FOR EACH ROW

3.14.3 Funzione da eseguire

check_storici

3.14.4 Descrizione

Questo trigger serve per impedire che vengano modificati o eliminati i dati degli studenti o degli esami nelle tabelle "Storico".