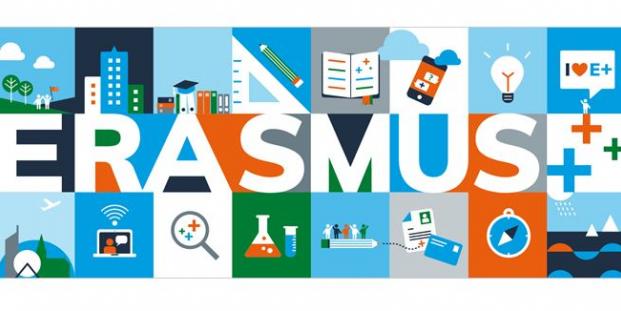
******

***DATABASE***

******

Andrea Ciccotti

Matricola: 0252497

Basi di Dati e Conoscenza A.A. 2018/2019

**Indice**

Introduzione 2

Analisi dei requisiti 2

Schema E-R logico 5

Schema E-R fisico 10

Creazione tabelle 13

Esempi inserimenti di prova nelle tabelle 16

Query 26

Algebra e Calcolo relazionale 37

Ottimizzazione 38

Indici 38

Trigger 41

Stored procedure e Cursori 45

Viste 47

MongoDB 48

**Introduzione**

L’**Erasmus**è un progetto dell’Unione Europea che permette agli studenti iscritti alle università europee di studiare in università di altri Paesi compresi nell’UE o a essa associati per un periodo che va dai 3 ai 12 mesi. Questa possibilità è data agli studenti delle lauree triennali e magistrali, ai laureandi che devono conseguire un tirocinio e a chi, già laureato, vuole effettuare all’estero il dottorato.

Il database in questione è stato progettato e realizzato con lo scopo di gestire il progetto Erasmus di ogni università.

Esso è strutturato in modo da voler soddisfare fondamentalmente i seguenti requisiti:

* Tenere sotto controllo le iscrizioni al progetto
* Gestire il percorso all’estero di ogni studente
* Ricavare informazioni statistiche quali numero di studenti partecipanti, università estere più scelte e numero di esami sostenuti durante il periodo all’estero

**Analisi requisiti**

**Università**: Istituto scolastico d'ordine superiore, che aderisce al programma Erasmus.

**Facoltà**: Unità didattica nella quale si raggruppano tutti gli insegnamenti appartenenti ad un determinato settore della scienza. L’insieme di queste compone l’università.

**Corso**: Corso di Laurea erogato da una facoltà.

**Nazione**: Stato in cui si trovano le varie università con i vari alloggi.

**Studente**: Colui che studia in una determinata università.

**Professore**: Colui che insegna in un’università.

**Alloggio**: Luogo dove gli studenti partecipanti all’Erasmus soggiornano durante il loro periodo di studi all’estero.

**Ufficio**: ­­Luogo presente in ogni facoltà dove richiedere informazioni e portare documenti riguardo l’Erasmus.

**Responsabile**: ­­Colui che è il responsabile dell’ufficio Erasmus.

**Graduatoria**: Classifica definitiva stilata da una commissione valutatrice secondo dei criteri, in base alla quale viene deciso chi è vincitore, chi è idoneo e chi è non idoneo delle borsa di studio Erasmus.

**Commissione**: Gruppo di professori istituito per ogni facoltà con il compito di riunirsi, valutare le domande di partecipazione e stilare la graduatoria Erasmus.

**Non Idoneo**: Tutti quei studenti che non hanno i requisiti richiesti per partecipare al progetto, e quindi vengono messi in fondo alla graduatoria.

**Idoneo**: Tutti quei studenti che nonostante abbiano i requisiti a causa della loro posizione in graduatoria in relazione al numero di posti disponibili non possono partecipare all’Erasmus a meno di rinunce o scorrimenti della graduatoria.

**Vincitore**: Tutti quei studenti che sono risultati vincitori della borsa di studio Erasmus e possono partire.

**Materia**: ­­Insegnamento erogato in un corso di studi di un’università da parte di un professore.

**Learning Agreement**: ­­È un accordo didattico bilaterale che prevede la scelta dell’università e degli esami da fare durante il periodo Erasmus, specificando la data di arrivo e di partenza.

Ogni Università situata in una determinata nazione è formata da delle facoltà, che offrono vari corsi di studio. In una nazione ci possono essere più università ma un’università può trovarsi solo in una nazione. La nazione d’appartenenza di un’università è importante per stabilire il contributo mensile Erasmus da dare a tutti quegli studenti che si recano in quel determinato paese. Ogni facoltà offre vari corsi di studi e un corso di studio è offerto solo da una facolta. In ciascuno corso di studio sono iscritti degli studenti, che possono essere della laurea triennale o magistrale. Ogni studente può essere iscritto solamente ad un corso di studi e in un corso di studi sono iscritti più studenti.

Qualsiasi studente iscritto all’università può o meno partecipare all’Erasmus. Per farlo, deve fare una domanda Erasmus che sarà valutata da una Commisione valutatrice. Quest’ultima, composta da professori, è istituità per ciascuna facoltà ed ha il compito di stilare la graduatoria Erasmus secondo dei criteri fissati. Ogni graduatoria è stilata da una commissione e una commissione può stilare varie graduatorie. In una graduatoria ci sono più studenti ma ogni studente fa parte di una solo graduatoria.

Ogni commissione fa parte di una facolta specifica e ogni facoltà può istituire più commissioni.

In base alla posizione in graduatoria, ciascun studente può risultare non idoneo, se non ha rispettato i criteri o se ha sbagliato la compilazione della domanda, idoneo, se nonostante abbia tutte le caratteristiche richiestà ha ottenuto un punteggio basso e a causa di un numero limitato di posti disponibili, e vincitore se ha ottenuto un punteggio alto ed è quindi rientrato tra i posti disponibili.

Tutti gli studenti vincitori devono quindi, prima di partire, compilare il Learning Agreement, che è un documento nel quale deve essere indicata: la facolta/corso di destinazione, la durata della mobilità e gli esami scelti da sostenere durante il periodo all’estero.

Ogni vincitore deve compilare un learning agreement ma lo stesso learning agreement può essere compilato da più studenti. In ogni learning agreement ci possono essere vari esami di materie e ogni materia può comparire come esame in learning agreement diversi. In ogni learning agreement va specificato il corso di destinazione scelto e lo stesso corso può essere scelto da learning agreement diversi.

Le facoltà/corsi all’estero scelti come destinazione devono aver stretto una collaborazione con il proprio corso di studi, stabilendo quindi il numero di posti disponibili e le competenze linguistiche richieste agli studenti. Ogni corso può collaborare con altri corsi.

Per ogni esame scelto si deve specificare il numero di crediti, l’esame corrispondente del proprio corso, e in caso di superamento il rispettivo voto.

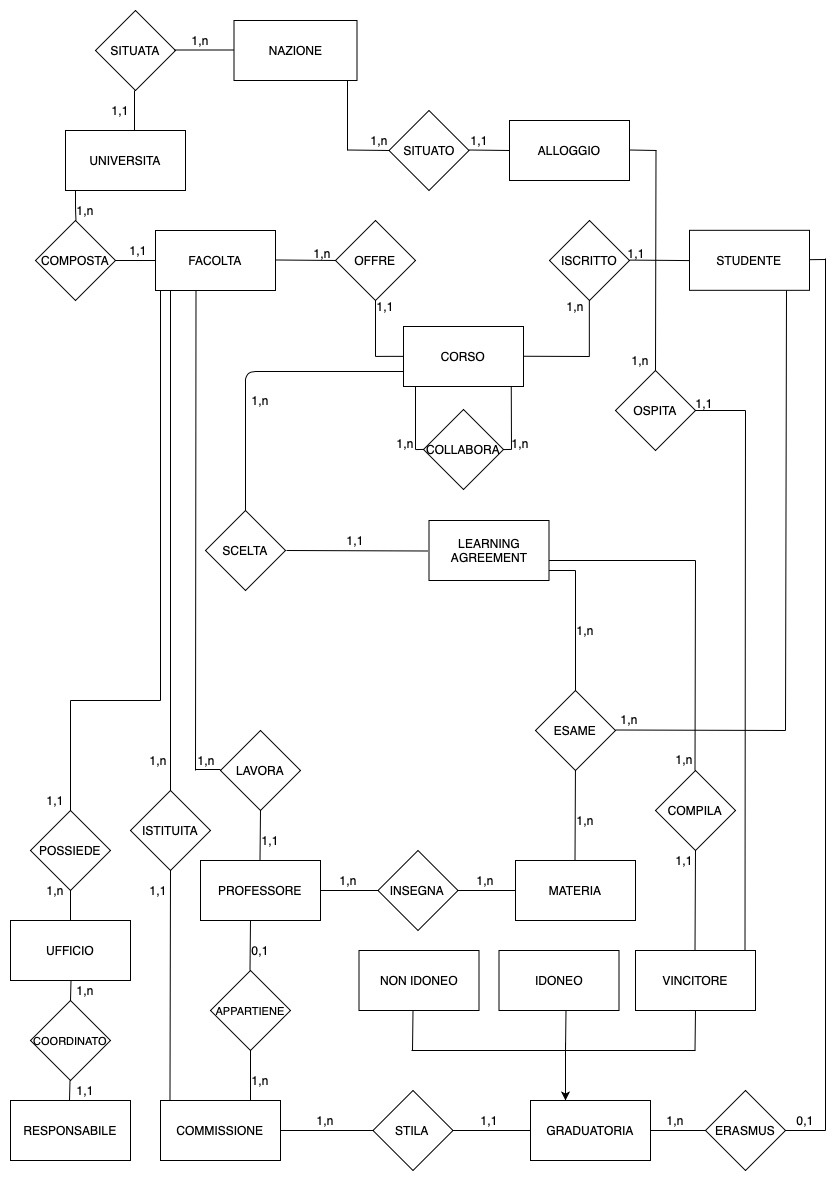
Gli studenti per qualsiasi dubbio o informazioni relativi al progetto Erasmus possono rivolgersi all’ufficio Erasmus della propria facoltà, presidiato da uno o più responsabili. Un ufficio può gestire gli scambi Erasmus di una o più facoltà e ognuna di queste ne ha uno. Ogni ufficio può essere cordinato da uno o più responsabili e un responsabile può gestire un solo ufficio.

In ogni facoltà lavorano dei professori, i quali insegnano una o più materie. Una materia a sua volta può essere insegnata da uno o più professori.

In una facoltà lavorano più professori ma un professore appartiene solamente ad una facoltà. Un professore può o meno essere membro di una commissione Erasmus, mentre una commissione Erasmus è di solito composta da più professori.

Ogni studente che parte in Erasmus deve trovarsi un alloggio (casa, stanza singola, hotel, ostello ecc) dove stare durante il periodo di soggiorno nella nazione estera. Ogni alloggio può ospitare uno o più studenti vincitori, ed è situato in una nazione. In ogni nazione ci sono più alloggi.

**Schema E-R logico**

**Nota**: Non ho inserito gli attributi direttamente nello schema E-R logico per renderlo più leggibile.

**Entità**

**UNIVERSITA**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Unicode | Int, Primary Key | Codice identificativo dell’università |
| Nome | Varchar | Nome dell’università |
| Indirizzo | Varchar | Indirizzo preciso dell’università |

**FACOLTA**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Nome | Varchar, Primary Key | Nome della facoltà |
| Sede | Varchar | Sede della facoltà |

**CORSO**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Codice | Varchar, Primary Key | Codice identificativo del corso |
| Nome | Varchar | Nome del corso |
| Tipologia | Varchar | Specifica se si tratta di un  corso triennale o  magistrale |

**STUDENTE**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| CF | Varchar, Primary Key | Codice fiscale identificativo dello studente |
| Matricola | Int | Numero di matricola dello studente |
| Nome | Varchar | Nome dello studente |
| Cognome | Varchar | Cognome dello studente |
| Data | Date | Data di nascita dello studente |
| Isee | Int | Valore Isee dello studente |
| CertLing | Varchar | Certificazioni linguistiche possedute dallo studente |
| Anno | Int | Specifica l’anno accademico a cui è iscritto lo studente |
| Media | Float | Media ponderata degli esami superati dallo studente |
| Mail | Varchar | Indirizzo mail dello studente |

**NAZIONE**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Nome | Varchar, Primary Key | Nome della nazione |
| Contributo | Int | Contributo mensile dato  ad ogni studente Erasmus |

**ALLOGGIO**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Nome | Varchar, Primary Key | Nome dell’alloggio |
| Sede | Varchar | Luogo dove si trova l’alloggio |
| Tipo | Varchar | Tipologia d’alloggio |
| Affitto | Int | Costo mensile dell’alloggio |
| Valutazione | Int | Recensione da 1 a 10 dell’alloggio |

**PROFESSORE**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| CF | Varchar, Primary Key | Codice fiscale del professore |
| Nome | Varchar | Nome del professore |
| Cognome | Varchar | Cognome del professore |
| Quallifica | Varchar | Qualifica del professore |
| Mail | Varchar | Indirizzo mail del professore |

**UFFICIO**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Sede | Varchar, Primary Key | Sede dell’ufficio |
| Telefono | Int | Numero di telefono dell’ufficio |
| OrarioA | Time | Orario apertura ufficio |
| OrarioC | Time | Orario chiusura ufficio |

**RESPONSABILE**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| CF | Varchar, Primary Key | Codice fiscale del responsabile dell’ufficio |
| Nome | Varchar | Nome del responsabile |
| Cognome | Varchar | Cognome del responsabile |
| Mail | Varchar | Indirizzo mail del responsabile |

**COMMISSIONE**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| ID | Int, Primary Key | Codice identificativo della commissione |
| Nome | Varchar | Nome della commissione |

**GRADUATORIA**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| ID | Int, Primary Key | Identificativo di ciascuna graduatoria |
| Posizione | Int, Primary Key | Posizione in graduatoria |
| Punteggio | Int | Punteggio di ciascun studente in graduatoria |
| Anno | Int | Anno di pubblicazione della graduatoria |

**MATERIA**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Nome | Varchar, Primary Key | Nome della Materia |
| CFU | Int | Numero di crediti della materia |

**LEARNING AGREEMENT**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Id | Int, Primary Key | Codice identificativo di ogni learning agreement |
| DataInizio | Date | Data in cui inizia la mobilità |
| DataFine | Date | Data in cui finisce la mobilità |

**Relazioni**

**COLLABORA**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Anno | Int | Anno della collaborazione |
| Posti | Int | Numero di posti disponibili per gli studenti Erasmus |
| Cert | Varchar | Le certificazioni linguistiche richieste |
| Durata | Int | Numero di mesi che dura il periodo di studio presso l’università estera |
| Livello | Int | Specifica se è una collaborazioni per una laurea triennale(1), magistrale/ciclo unico(2), un dottorato di ricerca (3), o per uno stage(4) |

**INSEGNA**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Anno | Int | Anno accademico di insegnamento |

**ESAME**

|  |  |  |
| --- | --- | --- |
| Nome | ATTRIBUTI  Tipo | Descrizione |
| Voto | Int | Voto dell’esame |
| Data | Int | Data di svolgimento dell’esame |

**Schema E-R fisico**

Dallo schema E-R logico ottengo il fisico attraverso:

1. **Eliminazione delle generalizzazioni**: nel database è presente una sola generalizzazione, dell’entità “GraduatoriaErasmus” nelle figlie “Vincitore”, “Idoneo” e “NonIdoneo”. Come modello di accorpamento ho deciso di utilizzare “Accorpamento delle figlie della generalizzazione nel padre”. Per fare ciò le tre entità figlie vengono eliminate e viene aggiunto un nuovo attributo “Risultato” di tipo Varchar che indica se si è idonei,non idonei o vincitori, all’entità “GraduatoriaErasmus”.
2. **Normalizzazione**: nel processo di normalizzazione vengono eliminate le relazioni inessenziali, ovvero quelle con cardinalità “1,N – 1,1” e sottoinsiemi. La normalizzazione consiste in tre fasi principali:
   * Cancellazione della relazione inessenziale.
   * Collegamento delle entità precedentemente relazionate da una freccia puntante l’entità che aveva cardinalità 1,N.
   * Inserimento nell’entità con precedente cardinalità 1,1 della Primary Key dell’entità puntata, sotto forma di Foreign Key.

Le **relazioni normalizzate** sono:

Università(1,N), Facoltà(1,1).

Nazione(1,N), Facoltà(1,1).

Nazione(1,N), Alloggio(1,1).

Alloggio(1,N), Vincitore(1,1).

Facoltà(1,N), Corso(1,1).

Corso(1,N), Studente(1,1).

Graduatoria (1,N), Studente(0,1).

Commissione (1,N), Professore(0,1).

Ufficio (1,N), Responsabile(1,1).

Graduatoria (1,N), Facoltà(1,1).

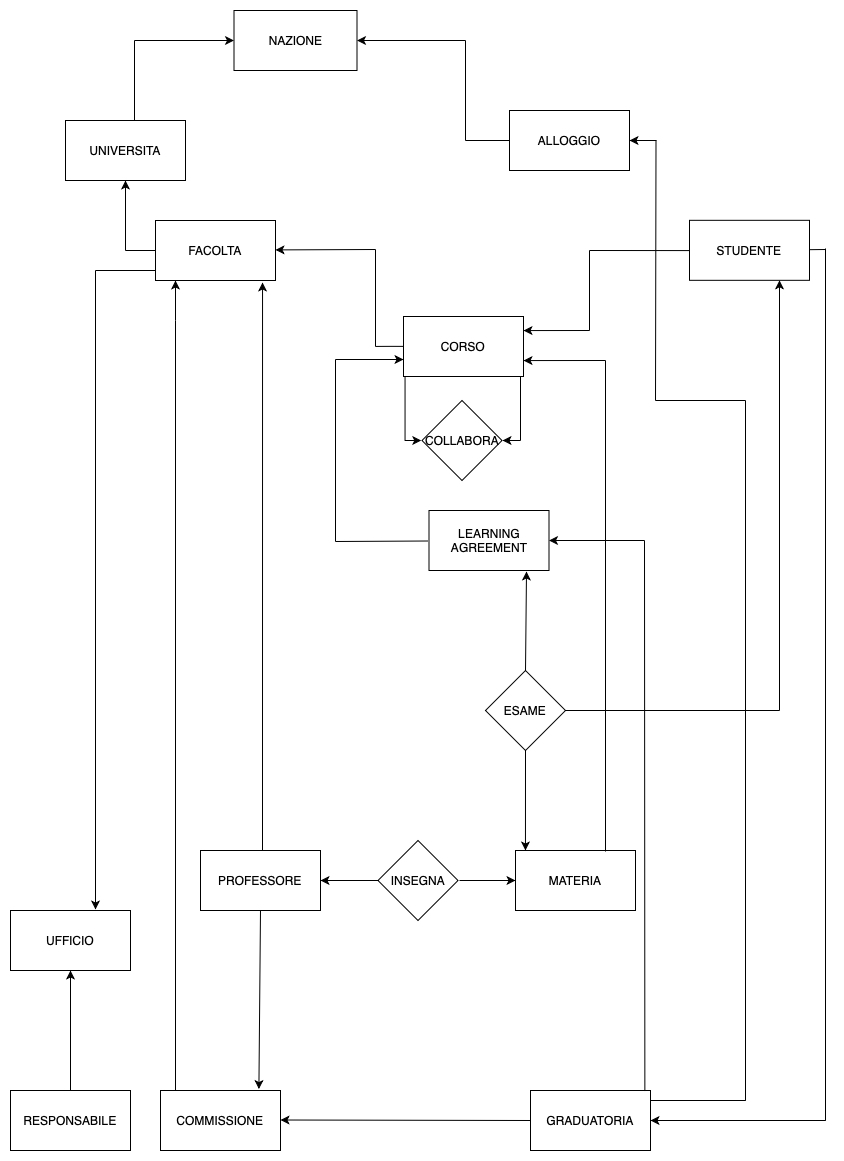
Ufficio (1,N), Facoltà(1,1).

Corso(1,N), LearningAgreement(1,1).

Facoltà(1,N), Professore(1,1).

Commissione(1,N), Graduatoria(1,1).

Le restanti relazioni nello schema E/R logico sono di cardinalità (1,N – 1,N), pertanto resteranno sullo schema normalizzato e diventeranno tabelle, successivamente, nella realizzazione fisica.



Le tabelle del database saranno quindi:

* NAZIONE(**Nome**, Contributo)
* UNIVERSITà(**Unicode**, Nome, Indirizzo, Nazione(FK))
* ALLOGGIO(**Nome**, Sede, Tipo, Affitto, Valutazione, Nazione(FK))
* UFFICIO (**Sede**, Telefono, OrarioA, OrarioC)
* RESPONSABILE(**CF**, Nome, Cognome, Mail, Ufficio(FK))
* FACOLTà(**Nome**, Sede, **Università(FK)**, UfficioErasmus(FK))
* CORSO(**Codice**, Nome, **Facoltà(FK)**, **Università(FK),** Tipologia)
* COLLABORA(**Corso1(FK), Facolta1(FK), Università1(FK), Corso2(FK), Facoltà2(FK), Università2(FK), Anno**, Posti, Cert, Durata, **Livello**)
* COMMISSIONE (**ID**, Nome, Facolta(FK),Università(FK))
* PROFESSORE(**CF**, Nome, Cognome, Qualifica, Mail, Facoltà(FK), Commissione(FK))
* MATERIA(**Nome**, CFU, **Corso(FK)**, **Facoltà(FK)**, **Università(FK)**)
* INSEGNA(**Professore(FK), Materia(FK), Corso(FK), Facoltà(FK), Università(FK), Anno**)
* LEARNINGAGREEMENT(**ID**, Corso(FK), Facolta(FK), Università(FK), DataInizio, DataFine)
* GRADUATORIA(**ID**, **Posizione**, Punteggio, Anno, Risultato, LearningAgreement(FK), Alloggio(FK), Commissione(FK))
* STUDENTE(**CF**, Matricola, Nome, Cognome, Data, Mail, Isee, CertLing, Media, Anno, Corso(FK), Facolta(FK), Universita(FK), Erasmus(FK), PosizioneErasmus(FK))
* ESAME(**Materia(FK),** Corso(FK),Facoltà(FK),Università(FK) **LearningAgreement(FK), Studente(FK), Voto, Data**)

**Nota:** Gli attributi in grassetto sono le chiavi primarie, mentre gli attributi seguiti dalla sigla FK sono le chiavi esterne.

**Creazione tabelle**

create database Erasmus;

use Erasmus;

create table NAZIONE(

Nome varchar(50) not null,

Contributo int not null,

Primary key (Nome)

) engine=INNODB;

create table UNIVERSITA (

Unicode int not null AUTO\_INCREMENT,

Nome varchar(50) not null,

Indirizzo varchar(50) not null,

Nazione varchar(50) not null,

primary key (Unicode),

foreign key (Nazione) references NAZIONE(Nome)

) engine=INNODB;

create table ALLOGGIO(

Nome varchar(50) not null,

Sede varchar(50) not null,

Tipo varchar(50) not null,

Affitto int not null,

Valutazione int not null,

Nazione varchar(50) not null,

primary key (Nome),

foreign key (Nazione) references NAZIONE(Nome)

) engine=INNODB;

create table UFFICIO(

Sede varchar(50) not null,

Telefono int not null,

OrarioA Time not null,

OrarioC Time not null,

primary key (Sede)

) engine=INNODB;

create table RESPONSABILE(

CF varchar(16) not null,

Nome varchar(50) not null,

Cognome varchar(50) not null,

Mail varchar(50) not null,

Ufficio varchar(50) not null,

primary key(CF),

foreign key(Ufficio) references UFFICIO(Sede)

) engine=INNODB;

create table FACOLTA(

Nome varchar(50) not null,

Sede varchar(50) not null,

Universita int not null,

Ufficio varchar(50) not null,

primary key(Nome, Universita),

foreign key(Universita) references UNIVERSITA(Unicode),

foreign key(Ufficio) references UFFICIO(Sede)

) engine=INNODB;

create table CORSO(

Codice varchar(50) not null,

Nome varchar(50) not null,

Facolta varchar(50) not null,

Universita int not null,

Tipologia varchar(50) not null,

primary key(Codice, Facolta, Universita),

foreign key(Facolta, Universita) references FACOLTA(Nome, Universita)

) engine=INNODB;

create table COLLABORA(

Corso1 varchar(50) not null,

Facolta1 varchar(50) not null,

Universita1 int not null,

Corso2 varchar(50) not null,

Facolta2 varchar(50) not null,

Universita2 int not null,

Anno int not null,

Posti int not null,

Cert varchar(50),

Durata int not null,

Livello int not null,

primary key(Corso1, Facolta1, Universita1, Corso2, Facolta2, Universita2, Anno, Livello),

foreign key(Corso1, Facolta1, Universita1) references CORSO(Codice, Facolta, Universita),

foreign key(Corso2, Facolta2, Universita2) references CORSO(Codice, Facolta, Universita)

) engine=INNODB;

create table COMMISSIONE(

ID int not null,

Nome varchar(50) not null,

Facolta varchar(50) not null,

Universita int not null,

primary key(ID),

foreign key(Facolta, Universita) references FACOLTA(Nome, Universita)

) engine=INNODB;

create table PROFESSORE(

CF varchar(16) not null,

Nome varchar(50) not null,

Cognome varchar(50) not null,

Qualifica varchar(50) not null,

Mail varchar(50) not null,

Facolta varchar(50) not null,

Universita int not null,

Commissione int,

primary key(CF),

foreign key(Facolta, Universita) references FACOLTA(Nome, Universita),

foreign key(Commissione) references COMMISSIONE(ID)

) engine=INNODB;

create table MATERIA(

Nome varchar(50) not null,

CFU int not null,

Corso varchar(50) not null,

Facolta varchar(50) not null,

Universita int not null,

primary key(Nome, Corso, Facolta, Universita),

foreign key(Corso, Facolta, Universita) references CORSO(Codice, Facolta, Universita)

) engine=INNODB;

create table INSEGNA(

Professore varchar(16) not null,

Materia varchar(50) not null,

Corso varchar(50) not null,

Facolta varchar(50) not null,

Universita int not null,

Anno int not null,

primary key(Professore, Materia, Corso, Facolta, Universita, Anno),

foreign key(Professore)references PROFESSORE(CF),

foreign key(Materia, Corso, Facolta, Universita) references MATERIA(Nome, Corso, Facolta, Universita)

) engine=INNODB;

create table LEARNINGAGREEMENT(

ID int not null,

Corso varchar(50) not null,

Facolta varchar(50) not null,

Universita int not null,

DataInizio Date,

DataFine Date,

primary key(ID),

foreign key(Corso, Facolta, Universita) references CORSO(Codice, Facolta, Universita)

) engine=INNODB;

create table GRADUATORIA(

ID int not null,

Posizione int not null,

Punteggio int not null,

Anno int not null,

Risultato varchar(50) not null,

LearningAgreement int,

Alloggio varchar(50),

Commissione int not null,

primary key(ID, Posizione),

foreign key(LearningAgreement) references LEARNINGAGREEMENT(ID),

foreign key(Alloggio) references ALLOGGIO(Nome),

foreign key(Commissione) references COMMISSIONE(ID)

) engine = INNODB;

create table STUDENTE(

CF varchar(16) not null,

Matricola varchar(7) not null,

Nome varchar(50) not null,

Cognome varchar(50) not null,

Data Date not null,

Mail varchar(50) not null,

Isee int not null,

CertLing varchar(50),

Media float not null,

Anno int not null,

Corso varchar(50) not null,

Facolta varchar(50) not null,

Universita int not null,

Erasmus int,

PosizioneErasmus int,

primary key(CF),

foreign key(Corso, Facolta, Universita) references CORSO(Codice, Facolta, Universita),

foreign key(Erasmus, PosizioneErasmus) references GRADUATORIA(ID, Posizione)

) engine = INNODB;

create table ESAME(

Materia varchar(50) not null,

Corso varchar(50) not null,

Facolta varchar(50) not null,

Universita int not null,

LearningAgreement int not null,

Studente varchar(50) not null,

Voto int not null,

Data Date,

primary key(Materia, LearningAgreement, Studente, Voto, Data),

foreign key(Materia, Corso, Facolta, Universita) references MATERIA(Nome, Corso, Facolta, Universita),

foreign key(LearningAgreement) references LEARNINGAGREEMENT(ID),

foreign key(Studente) references STUDENTE(CF)

) engine = INNODB;

**Inserimenti di record nelle tabelle**

Insert into NAZIONE(Nome, Contributo) values ('Spagna',250);

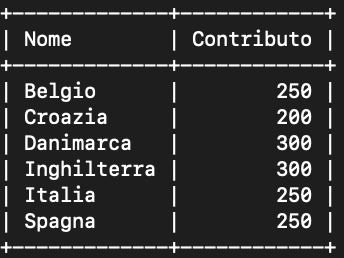
Insert into NAZIONE(Nome, Contributo) values ('Belgio',250);

Insert into NAZIONE(Nome, Contributo) values ('Italia',250);

Insert into NAZIONE(Nome, Contributo) values ('Danimarca',300);

Insert into NAZIONE(Nome, Contributo) values ('Croazia',200);

Insert into NAZIONE(Nome, Contributo) values ('Inghilterra',300);



Insert into UNIVERSITA(Unicode, Nome, Indirizzo, Nazione) values (1, 'Università di Roma Tor Vergata', 'Via Cracovia 50', 'Italia');

Insert into UNIVERSITA(Unicode, Nome, Indirizzo, Nazione) values (2, 'Università di Granada', 'Av. Del Hospicio', 'Spagna');

Insert into UNIVERSITA(Unicode, Nome, Indirizzo, Nazione) values (3, 'Università di Roma La Sapienza', 'Piazzale Aldo Moro 5', 'Italia');

Insert into UNIVERSITA(Unicode, Nome, Indirizzo, Nazione) values (4, 'Università di Anversa', 'Prinsstraat 13', 'Belgio');

Insert into UNIVERSITA(Unicode, Nome, Indirizzo, Nazione) values (5, 'Università di Copenhagen', 'Norregade 10', 'Danimarca');

Insert into UNIVERSITA(Unicode, Nome, Indirizzo, Nazione) values (6, 'Università di Zagabria', 'Trg Republike Hrvatske 14', 'Croazia');



Insert into ALLOGGIO(Nome, Sede, Tipo, Affitto, Valutazione, Nazione) values ('CampusX Roma', 'Via di Passo Lombardo 341', 'Resort per gli studenti', 500, 9, 'Italia');

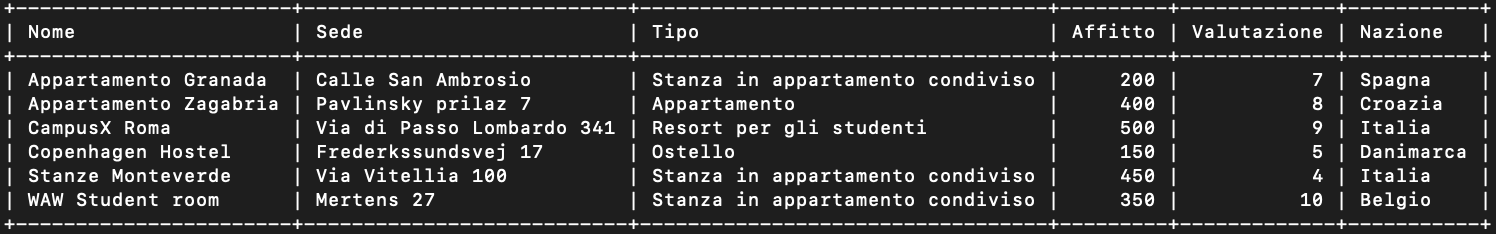
Insert into ALLOGGIO(Nome, Sede, Tipo, Affitto, Valutazione, Nazione) values ('Appartamento Granada', 'Calle San Ambrosio', 'Stanza in appartamento condiviso', 200, 7, 'Spagna');

Insert into ALLOGGIO(Nome, Sede, Tipo, Affitto, Valutazione, Nazione) values ('Appartamento Zagabria', 'Pavlinsky prilaz 7', 'Appartamento', 400, 8, 'Croazia');

Insert into ALLOGGIO(Nome, Sede, Tipo, Affitto, Valutazione, Nazione) values ('Copenhagen Hostel', 'Frederkssundsvej 17', 'Ostello', 150, 5, 'Danimarca');

Insert into ALLOGGIO(Nome, Sede, Tipo, Affitto, Valutazione, Nazione) values ('WAW Student room', 'Mertens 27', 'Stanza in appartamento condiviso', 350, 10, 'Belgio');

Insert into ALLOGGIO(Nome, Sede, Tipo, Affitto, Valutazione, Nazione) values ('Stanze Monteverde', 'Via Vitellia 100', 'Stanza in appartamento condiviso', 450, 4, 'Italia');



Insert into UFFICIO(Sede, Telefono, OrarioA, OrarioC) values ('Via della Ricerca Scientifica 1', 0672594498, '09:00:00', '13:00:00');

Insert into UFFICIO(Sede, Telefono, OrarioA, OrarioC) values ('Via Columbia 2', 0672595836, '09:00:00', '13:00:00');

Insert into UFFICIO(Sede, Telefono, OrarioA, OrarioC) values ('Via Cracovia', 0672594002, '09:00:00', '13:00:00');

Insert into UFFICIO(Sede, Telefono, OrarioA, OrarioC) values ('Via del Politecnico 1', 0672597599, '09:00:00', '16:00:00');

Insert into UFFICIO(Sede, Telefono, OrarioA, OrarioC) values ('Piazzale Aldo Moro 5', 0643415062, '09:00:00', '11:00:00');

Insert into UFFICIO(Sede, Telefono, OrarioA, OrarioC) values ('Norregade 10', 123456789, '14:00:00', '18:00:00');

Insert into UFFICIO(Sede, Telefono, OrarioA, OrarioC) values ('Av. Del Hospicio', 11111111, '14:00:00', '20:00:00');



Insert into RESPONSABILE(CF, Nome, Cognome, Mail, Ufficio) values ('WKLRLD71C26B062J', 'Walter', 'Erisken', 'waltererisken@gmail.com', 'Norregade 10');

Insert into RESPONSABILE(CF, Nome, Cognome, Mail, Ufficio) values ('JLCCCW29T42I234Y', 'Jane', 'Ciano', 'jeaneciano@gmail.com', 'Via Cracovia');

Insert into RESPONSABILE(CF, Nome, Cognome, Mail, Ufficio) values ('DDHRQX85M06D092S', 'Davide', 'Bianchi', 'davidebianchi@gmail.com', 'Via Cracovia');

Insert into RESPONSABILE(CF, Nome, Cognome, Mail, Ufficio) values ('ZDOPPL97A18I120W', 'Zanna', 'Davide', 'davidebianchi@gmail.com', 'Piazzale Aldo Moro 5');

Insert into RESPONSABILE(CF, Nome, Cognome, Mail, Ufficio) values ('BLCZBJ57P14A038C', 'Gabriella', 'Pallicca', 'gabriellapallicca@libero.it', 'Via del Politecnico 1');

Insert into RESPONSABILE(CF, Nome, Cognome, Mail, Ufficio) values ('LCVFBF78B17F704W', 'Luisa', 'Bucci', 'luisabucci@uniroma2.it', 'Via della Ricerca Scientifica 1');

Insert into RESPONSABILE(CF, Nome, Cognome, Mail, Ufficio) values ('RXFJVN96H47C508T', 'Roberto', 'Moreno', 'robertomoreno@unigranada.com', 'Av. Del Hospicio');



Insert into FACOLTA(Nome, Sede, Universita, Ufficio) values ('Facolta di Economia', 'Via Columbia 2', 1, 'Via Columbia 2');

Insert into FACOLTA(Nome, Sede, Universita, Ufficio) values ('Facolta di Giurisprudenza', 'Via Cracovia', 1, 'Via Cracovia');

Insert into FACOLTA(Nome, Sede, Universita, Ufficio) values ('Facolta di Ingegneria', 'Via del Politecnico 1', 1, 'Via del Politecnico 1');

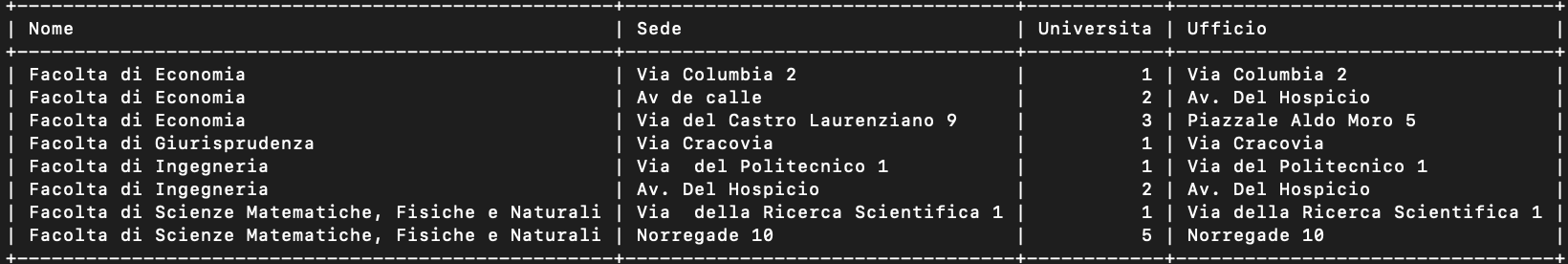
Insert into FACOLTA(Nome, Sede, Universita, Ufficio) values ('Facolta di Scienze Matematiche, Fisiche e Naturali', 'Via della Ricerca Scientifica 1', 1, 'Via della Ricerca Scientifica 1');

Insert into FACOLTA(Nome, Sede, Universita, Ufficio) values ('Facolta di Scienze Matematiche, Fisiche e Naturali', 'Norregade 10', 5, 'Norregade 10');

Insert into FACOLTA(Nome, Sede, Universita, Ufficio) values ('Facolta di Ingegneria', 'Av. Del Hospicio', 2, 'Av. Del Hospicio');

Insert into FACOLTA(Nome, Sede, Universita, Ufficio) values ('Facolta di Economia', 'Via del Castro Laurenziano 9', 3, 'Piazzale Aldo Moro 5');

Insert into FACOLTA(Nome, Sede, Universita, Ufficio) values ('Facolta di Economia', 'Av de calle', 2, 'Av. Del Hospicio');



Insert into CORSO(Codice, Nome, Facolta, Universita, Tipologia) values ('E01', 'Economia e Finanza', 'Facoltà di Economia', 1, 'Corso di laurea triennale');

Insert into CORSO(Codice, Nome, Facolta, Universita, Tipologia) values ('E02', 'Economia e Management', 'Facoltà di Economia', 1, 'Corso di laurea triennale');

Insert into CORSO(Codice, Nome, Facolta, Universita, Tipologia) values ('B01', 'Business administration and economics', 'Facoltà di Economia', 2, 'Corso di laurea triennale');

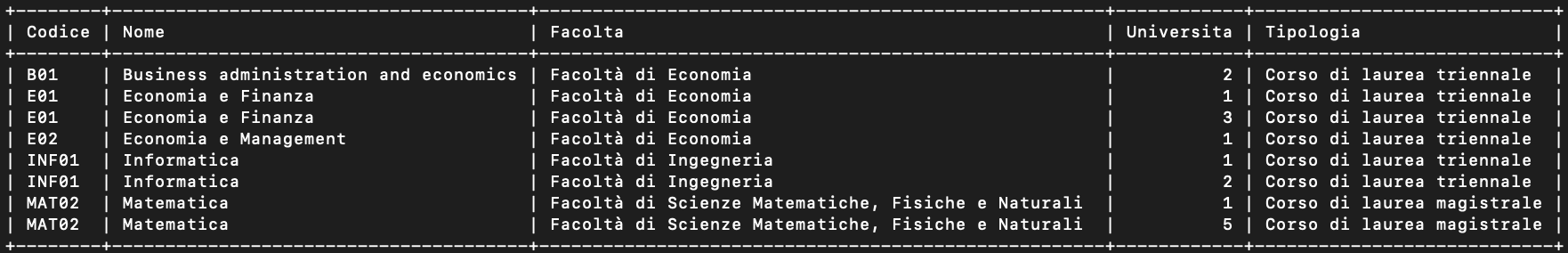
Insert into CORSO(Codice, Nome, Facolta, Universita, Tipologia) values ('MAT02', 'Matematica', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 1, 'Corso di laurea magistrale');

Insert into CORSO(Codice, Nome, Facolta, Universita, Tipologia) values ('MAT02', 'Matematica', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 5, 'Corso di laurea magistrale');

Insert into CORSO(Codice, Nome, Facolta, Universita, Tipologia) values ('E01','Economia e Finanza', 'Facoltà di Economia', 3, 'Corso di laurea triennale');

Insert into CORSO(Codice, Nome, Facolta, Universita, Tipologia) values ('INF01','Informatica', 'Facoltà di Ingegneria', 1, 'Corso di laurea triennale');

Insert into CORSO(Codice, Nome, Facolta, Universita, Tipologia) values ('INF01','Informatica', 'Facoltà di Ingegneria', 2, 'Corso di laurea triennale');



Insert into COLLABORA(Corso1, Facolta1, Universita1, Corso2, Facolta2, Universita2, Anno, Posti, Cert, Durata, Livello) values ('MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 1, 'MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 5, 2019, 2, 'B2 Inglese', 6, 2);

Insert into COLLABORA(Corso1, Facolta1, Universita1, Corso2, Facolta2, Universita2, Anno, Posti, Cert, Durata, Livello) values ('MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 1, 'MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 5, 2017, 2, 'B2 Inglese', 6, 2);

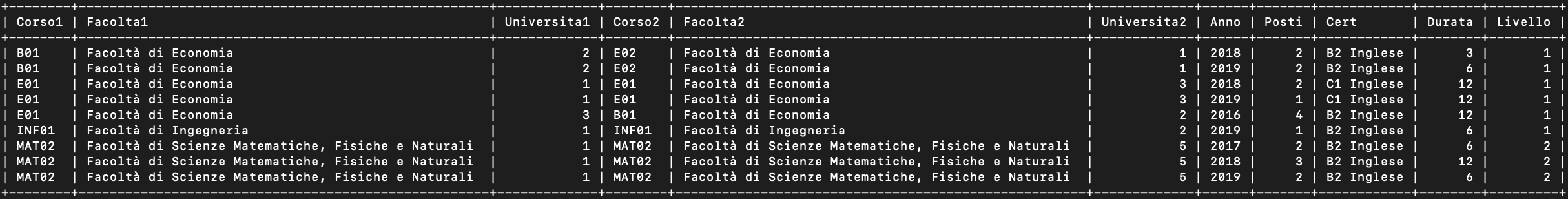
Insert into COLLABORA(Corso1, Facolta1, Universita1, Corso2, Facolta2, Universita2, Anno, Posti, Cert, Durata, Livello) values ('MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 1, 'MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 5, 2018, 3, 'B2 Inglese', 12, 2);

Insert into COLLABORA(Corso1, Facolta1, Universita1, Corso2, Facolta2, Universita2, Anno, Posti, Cert, Durata, Livello) values ('B01', 'Facoltà di Economia', 2, 'E02', 'Facoltà di Economia', 1, 2019, 2, 'B2 Inglese', 6, 1);

Insert into COLLABORA(Corso1, Facolta1, Universita1, Corso2, Facolta2, Universita2, Anno, Posti, Cert, Durata, Livello) values ('B01', 'Facoltà di Economia', 2, 'E02', 'Facoltà di Economia', 1, 2018, 2, 'B2 Inglese', 3, 1);

Insert into COLLABORA(Corso1, Facolta1, Universita1, Corso2, Facolta2, Universita2, Anno, Posti, Cert, Durata, Livello) values ('E01', 'Facoltà di Economia', 3, 'B01', 'Facoltà di Economia', 2, 2016, 4, 'B2 Inglese', 12, 1);

Insert into COLLABORA(Corso1, Facolta1, Universita1, Corso2, Facolta2, Universita2, Anno, Posti, Cert, Durata, Livello) values ('INF01', 'Facoltà di Ingegneria', 1, 'INF01', 'Facoltà di Ingegneria', 2, 2019, 1, 'B2 Inglese', 6, 1);



Insert into COMMISSIONE(ID, Nome , Facolta, Universita) values (1, 'Commissione Economia Erasmus', 'Facoltà di Economia', 1);

Insert into COMMISSIONE(ID, Nome , Facolta, Universita) values (2, 'Commissione Economia Erasmus', 'Facoltà di Economia', 1);

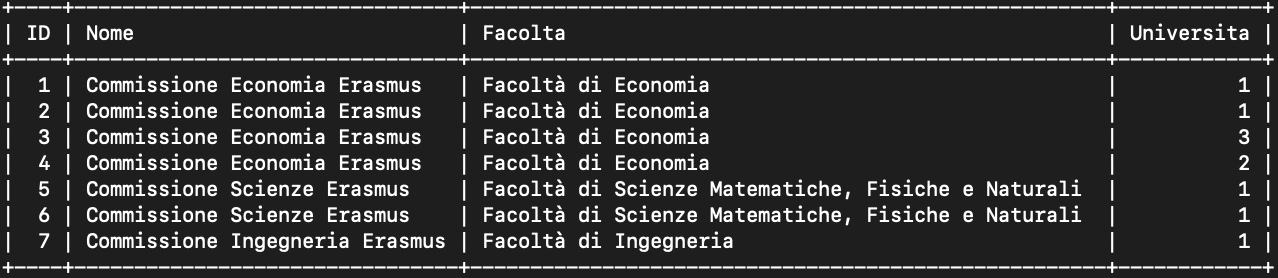
Insert into COMMISSIONE(ID, Nome , Facolta, Universita) values (3, 'Commissione Economia Erasmus', 'Facoltà di Economia', 3);

Insert into COMMISSIONE(ID, Nome , Facolta, Universita) values (4, 'Commissione Economia Erasmus', 'Facoltà di Economia', 2);

Insert into COMMISSIONE(ID, Nome , Facolta, Universita) values (5, 'Commissione Scienze Erasmus', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 1);

Insert into COMMISSIONE(ID, Nome , Facolta, Universita) values (6, 'Commissione Scienze Erasmus', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 1);

Insert into COMMISSIONE(ID, Nome , Facolta, Universita) values (7, 'Commissione Ingegneria Erasmus', 'Facoltà di Ingegneria, 1);



Insert into PROFESSORE(CF, Nome, Cognome, Qualifica, Mail, Facolta, Universita, Commissione) values ('BZDMRA85H16A735M', 'Mario', 'Rossi', 'Ordinario', 'mariorossi@uniroma2.com', 'Facoltà di Economia', '1', 1);

Insert into PROFESSORE(CF, Nome, Cognome, Qualifica, Mail, Facolta, Universita, Commissione) values ('NPCBHL95H15I668S', 'Enzo', 'Ferrari', 'Associato', 'enzoferrarii@uniroma1.com', 'Facoltà di Economia', '3', 3);

Insert into PROFESSORE(CF, Nome, Cognome, Qualifica, Mail, Facolta, Universita, Commissione) values ('VNQPSK48E57Z117B', 'Giuseppe', 'Bianchi', 'Ordinario', 'giuseppebianchi@unigranada.com', 'Facoltà di Economia', '2', 4);

Insert into PROFESSORE(CF, Nome, Cognome, Qualifica, Mail, Facolta, Universita, Commissione) values ('CYNTTB76B23H556Z', 'Andrea', 'Gallo', 'Ordinario', 'andreagallo@uniroma2.com', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', '1', 1);

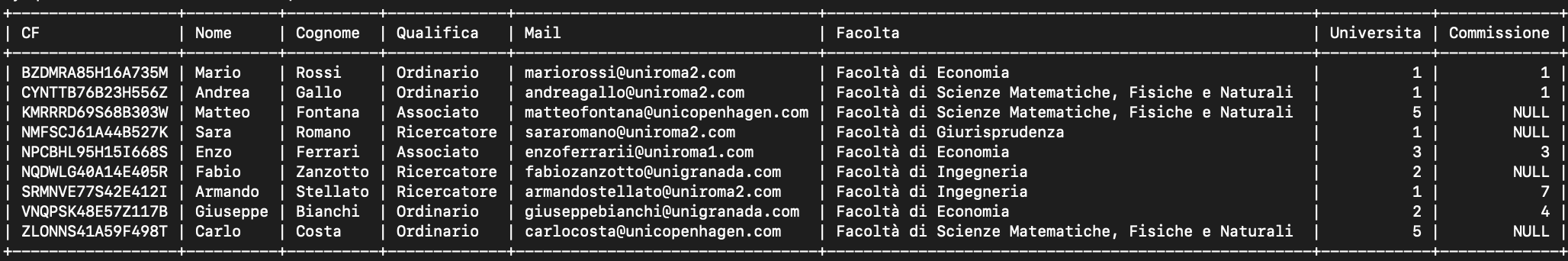
Insert into PROFESSORE(CF, Nome, Cognome, Qualifica, Mail, Facolta, Universita, Commissione) values ('KMRRRD69S68B303W', 'Matteo', 'Fontana', 'Associato', 'matteofontana@unicopenhagen.com', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', '5', NULL);

Insert into PROFESSORE(CF, Nome, Cognome, Qualifica, Mail, Facolta, Universita, Commissione) values ('ZLONNS41A59F498T', 'Carlo', 'Costa', 'Ordinario', 'carlocosta@unicopenhagen.com', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', '5', NULL);

Insert into PROFESSORE(CF, Nome, Cognome, Qualifica, Mail, Facolta, Universita, Commissione) values ('NMFSCJ61A44B527K', 'Sara', 'Romano', 'Ricercatore', 'sararomano@uniroma2.com', 'Facoltà di Giurisprudenza', '1', NULL);

Insert into PROFESSORE(CF, Nome, Cognome, Qualifica, Mail, Facolta, Universita, Commissione) values ('NQDWLG40A14E405R', 'Fabio', 'Zanzotto', 'Ricercatore', 'fabiozanzotto@unigranada.com', 'Facoltà di Ingegneria', '2', NULL);

Insert into PROFESSORE(CF, Nome, Cognome, Qualifica, Mail, Facolta, Universita, Commissione) values ('SRMNVE77S42E412I', 'Armando', 'Stellato', 'Ricercatore', 'armandostellato@uniroma2.com', 'Facoltà di Ingegneria', '1', 7);



Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Diritto pubblico', 12, 'E01', 'Facoltà di Economia', 1);

Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Analisi 1', 12, 'MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 5);

Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Analisi 1', 12, 'MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 1);

Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Geometria', 6, 'MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 1);

Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Algebra', 9, 'MAT02', 'Facoltà di Scienze Matematiche, Fisiche e Naturali', 5);

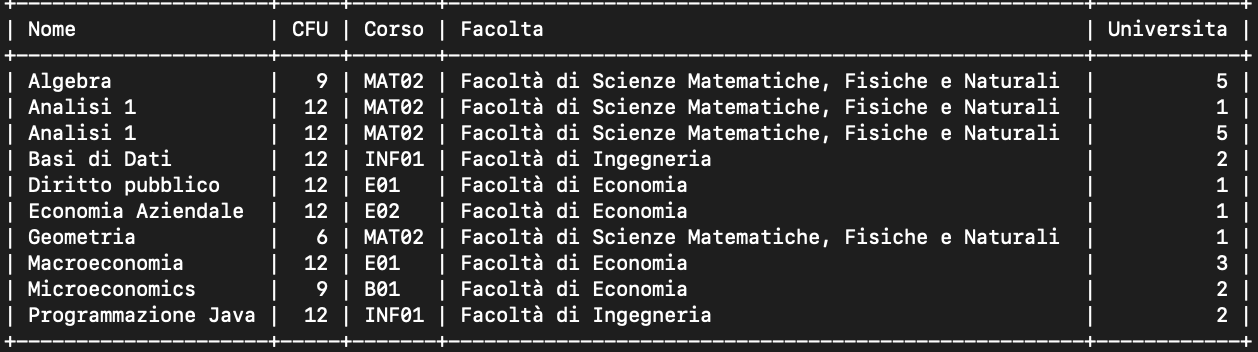
Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Economia Aziendale', 12, 'E02', 'Facoltà di Economia', 1);

Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Macroeconomia', 12, 'E01', 'Facoltà di Economia', 3);

Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Microeconomics', 9, 'B01', 'Facoltà di Economia', 2);

Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Programmazione Java', 12, 'INF01', 'Facoltà di Ingegneria', 2);

Insert into MATERIA(Nome, CFU, Corso, Facolta, Universita) values ('Basi di Dati', 12, 'INF01', 'Facoltà di Ingegneria', 2);



Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('ZLONNS41A59F498T', 'Analisi 1', 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 5, 2019);

Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('VNQPSK48E57Z117B', 'Microeconomics', 'B01', 'Facolta di Economia', 2, 2019);

Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('NPCBHL95H15I668S', 'Macroeconomia', 'E01', 'Facolta di Economia', 3, 2018);

Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('KMRRRD69S68B303W', 'Algebra', 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 5, 2019);

Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('NPCBHL95H15I668S', 'Macroeconomia', 'E01', 'Facolta di Economia', 3, 2019);

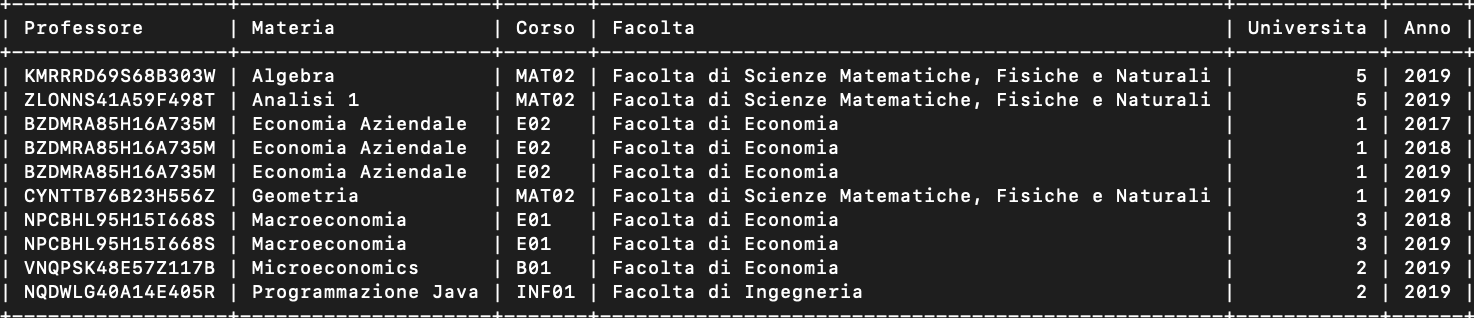
Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('BZDMRA85H16A735M', 'Economia Aziendale', 'E02', 'Facolta di Economia', 1, 2018);

Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('BZDMRA85H16A735M', 'Economia Aziendale', 'E02', 'Facolta di Economia', 1, 2017);

Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('BZDMRA85H16A735M', 'Economia Aziendale', 'E02', 'Facolta di Economia', 1, 2019);

Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('CYNTTB76B23H556Z', 'Geometria', 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 1, 2019);

Insert into INSEGNA(Professore, Materia, Corso, Facolta, Universita, Anno) values ('NQDWLG40A14E405R', 'Programmazione Java', 'INF01', 'Facolta di Ingegneria', 2, 2019);



Insert into LEARNINGAGREEMENT(ID, Corso, Facolta, Universita, DataInizio, DataFine) values (1, 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 5, '2019-09-01', '2020-03-01');

Insert into LEARNINGAGREEMENT(ID, Corso, Facolta, Universita, DataInizio, DataFine) values (2, 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 5, '2019-09-01', '2020-03-01');

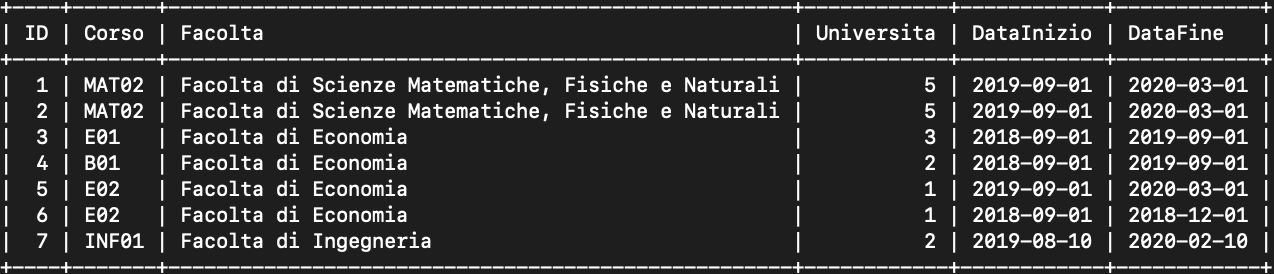
Insert into LEARNINGAGREEMENT(ID, Corso, Facolta, Universita, DataInizio, DataFine) values (3, 'E01', 'Facolta di Economia', 3, '2018-09-01', '2019-09-01');

Insert into LEARNINGAGREEMENT(ID, Corso, Facolta, Universita, DataInizio, DataFine) values (4, 'B01', 'Facolta di Economia', 2, '2018-09-01', '2019-09-01');

Insert into LEARNINGAGREEMENT(ID, Corso, Facolta, Universita, DataInizio, DataFine) values (5, 'E02', 'Facolta di Economia', 1, '2019-09-01', '2020-03-01');

Insert into LEARNINGAGREEMENT(ID, Corso, Facolta, Universita, DataInizio, DataFine) values (6, 'E02', 'Facolta di Economia', 1, '2018-09-01', '2018-12-01');

Insert into LEARNINGAGREEMENT(ID, Corso, Facolta, Universita, DataInizio, DataFine) values (7, 'INF01', 'Facolta di Ingegneria', 2, '2019-08-10', '2020-02-10');



Insert into GRADUATORIA(ID, Posizione, Punteggio, Anno, Risultato, LearningAgreement, Alloggio, Commissione) values(1, 1, 100, 2019, 'Vincitore', 1, 'Copenhagen Hostel', 5);

Insert into GRADUATORIA(ID, Posizione, Punteggio, Anno, Risultato, LearningAgreement, Alloggio, Commissione) values(1, 2, 98, 2019, 'Vincitore', 2, 'Copenhagen Hostel', 5);

Insert into GRADUATORIA(ID, Posizione, Punteggio, Anno, Risultato, LearningAgreement, Alloggio, Commissione) values(1, 78, 30, 2019, 'Idoneo', NULL, NULL, 5);

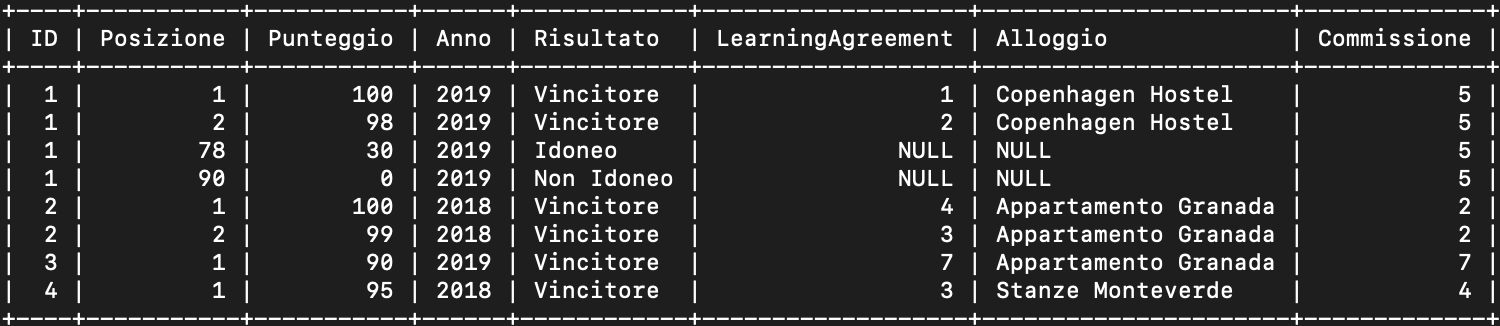
Insert into GRADUATORIA(ID, Posizione, Punteggio, Anno, Risultato, LearningAgreement, Alloggio, Commissione) values(1, 90, 0, 2019, 'Non Idoneo', NULL, NULL, 5);

Insert into GRADUATORIA(ID, Posizione, Punteggio, Anno, Risultato, LearningAgreement, Alloggio, Commissione) values(2, 1, 100, 2018, 'Vincitore', 4, 'Appartamento Granada', 2);

Insert into GRADUATORIA(ID, Posizione, Punteggio, Anno, Risultato, LearningAgreement, Alloggio, Commissione) values(2, 2, 99, 2018, 'Vincitore', 3, 'Appartamento Granada', 2);

Insert into GRADUATORIA(ID, Posizione, Punteggio, Anno, Risultato, LearningAgreement, Alloggio, Commissione) values(3, 1, 90, 2019, 'Vincitore', 7, 'Appartamento Granada', 7);

Insert into GRADUATORIA(ID, Posizione, Punteggio, Anno, Risultato, LearningAgreement, Alloggio, Commissione) values(4, 1, 95, 2018, 'Vincitore', 3, 'Stanze Monteverde’, 4);



Insert into STUDENTE(CF, Matricola, Nome, Cognome, Data, Mail, Isee, CertLing, Media, Anno, Corso, Facolta, Universita, Erasmus, PosizioneErasmus) values ('CCCNDR98P07H501O', '0252497', 'Andrea', 'Ciccotti', '1998-09-07', 'ciccoandrea.98@gmail.com', 30000, 'B2 inglese', 25.60, 2, 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 1, 1, 2);

Insert into STUDENTE(CF, Matricola, Nome, Cognome, Data, Mail, Isee, CertLing, Media, Anno, Corso, Facolta, Universita, Erasmus, PosizioneErasmus) values ('RCGTJJ43A45D208B', '0272400', 'Roberta', 'Trovato', '1997-01-17', 'robertatrovato@gmail.com', 17000, 'C1 inglese', 29, 3, 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 1, NULL, NULL);

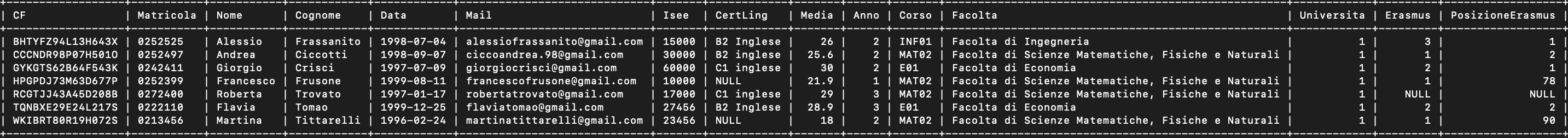
Insert into STUDENTE(CF, Matricola, Nome, Cognome, Data, Mail, Isee, CertLing, Media, Anno, Corso, Facolta, Universita, Erasmus, PosizioneErasmus) values ('GYKGTS62B64F543K', '0242411', 'Giorgio', 'Crisci', '1997-07-9', 'giorgiocrisci@gmail.com', 60000, 'C1 inglese', 30, 2, 'E01', 'Facolta di Economia', 1, 2, 1);

Insert into STUDENTE(CF, Matricola, Nome, Cognome, Data, Mail, Isee, CertLing, Media, Anno, Corso, Facolta, Universita, Erasmus, PosizioneErasmus) values ('TQNBXE29E24L217S', '0222110', 'Flavia', 'Tomao', '1999-12-25', 'flaviatomao@gmail.com', 27456, 'B2 Inglese', 28.9, 3, 'B01', 'Facolta di Economia', 2, 4, 1);

Insert into STUDENTE(CF, Matricola, Nome, Cognome, Data, Mail, Isee, CertLing, Media, Anno, Corso, Facolta, Universita, Erasmus, PosizioneErasmus) values ('HPGPDJ73M63D677P', '0252399', 'Francesco', 'Frusone', '1999-08-11', 'francescofrusone@gmail.com', 10000, NULL, 21.9, 1, 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 1, 1, 78);

Insert into STUDENTE(CF, Matricola, Nome, Cognome, Data, Mail, Isee, CertLing, Media, Anno, Corso, Facolta, Universita, Erasmus, PosizioneErasmus) values ('WKIBRT80R19H072S', '0213456', 'Martina', 'Tittarelli', '1996-02-24', 'martinatittarelli@gmail.com', 23456, NULL, 18, 2, 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 1, 1, 90);

Insert into STUDENTE(CF, Matricola, Nome, Cognome, Data, Mail, Isee, CertLing, Media, Anno, Corso, Facolta, Universita, Erasmus, PosizioneErasmus) values ('BHTYFZ94L13H643X', '0252525', 'Alessio', 'Frassanito', '1998-07-04', 'alessiofrassanito@gmail.com', 15000, 'B2 Inglese', 26, 2, 'INF01', 'Facolta di Ingegneria', 1, 3, 1);



Insert into ESAME(Materia, Corso, Facolta, Universita, LearningAgreement, Studente, Voto, Data) values ('Algebra', 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 5, 1, 'CCCNDR98P07H501O', 30, '2019-10-07');

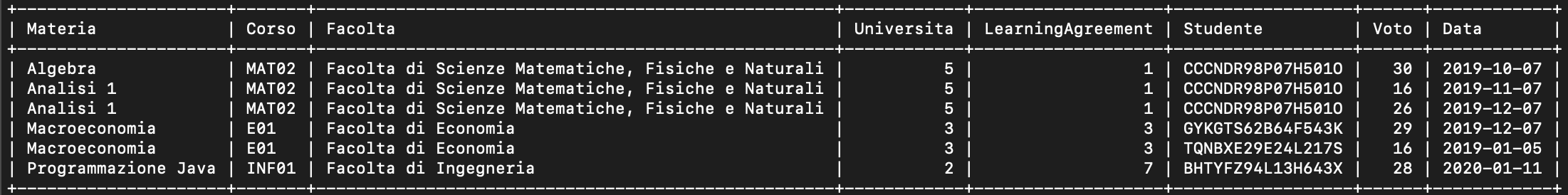
Insert into ESAME(Materia, Corso, Facolta, Universita, LearningAgreement, Studente, Voto, Data) values ('Analisi 1', 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 5, 1, 'CCCNDR98P07H501O', 16, '2019-11-07');

Insert into ESAME(Materia, Corso, Facolta, Universita, LearningAgreement, Studente, Voto, Data) values ('Analisi 1', 'MAT02', 'Facolta di Scienze Matematiche, Fisiche e Naturali', 5, 1, 'CCCNDR98P07H501O', 26, '2019-12-07');

Insert into ESAME(Materia, Corso, Facolta, Universita, LearningAgreement, Studente, Voto, Data) values ('Macroeconomia', 'E01', 'Facolta di Economia', 3, 3, 'GYKGTS62B64F543K', 29, '2019-12-07');

Insert into ESAME(Materia, Corso, Facolta, Universita, LearningAgreement, Studente, Voto, Data) values ('Programmazione Java', 'INF01', 'Facolta di Ingegneria', 2, 7, 'BHTYFZ94L13H643X', 28, '2020-01-11');

Insert into ESAME(Materia, Corso, Facolta, Universita, LearningAgreement, Studente, Voto, Data) values ('Macroeconomia', 'E01', 'Facolta di Economia', 3, 3, 'TQNBXE29E24L217S', 16, '2019-01-05');



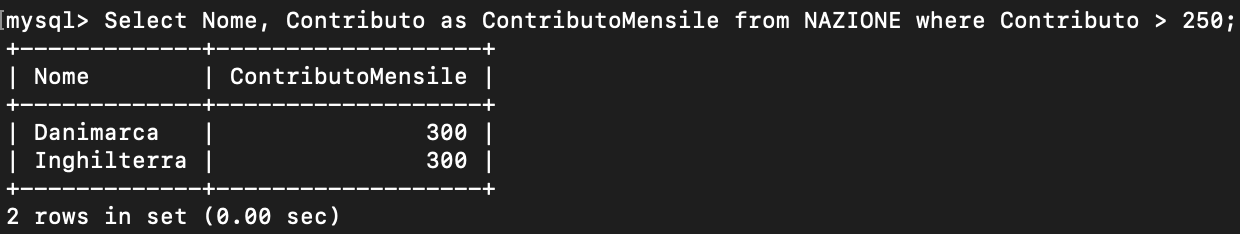
**Query**

1. **Selezionare le nazioni con contributo mensile > 250.**

Select Nome, Contributo as ContributoMensile

from NAZIONE

where Contributo > 250;

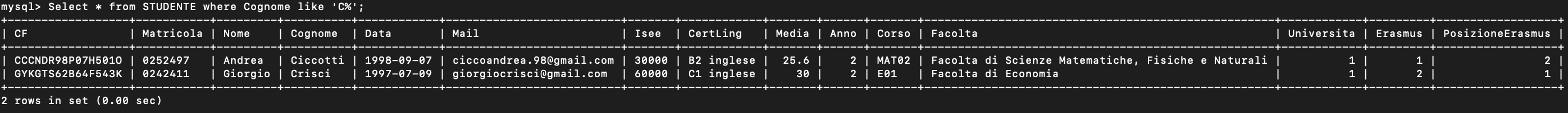


1. **Selezionare gli studenti che hanno il cognome che inizia per ‘C’.**

Select \*

from STUDENTE

where Cognome like 'C%';

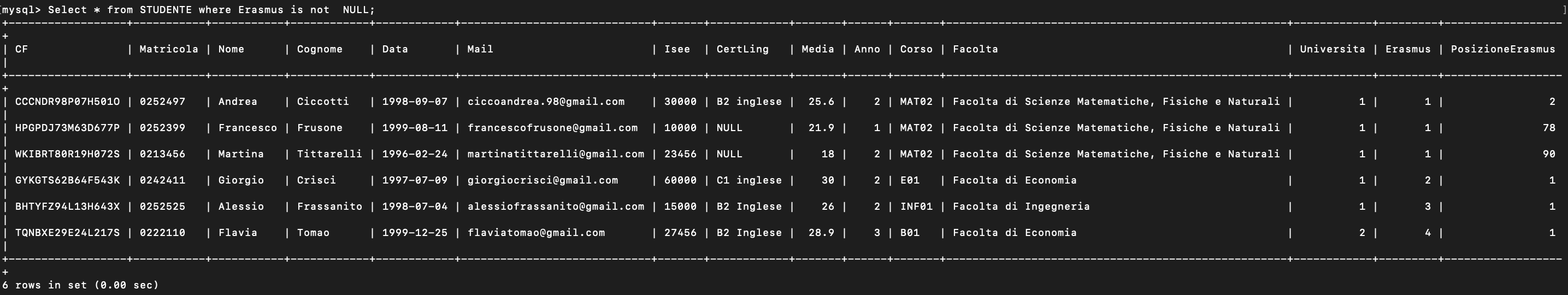


1. **Selezionare tutti gli studenti che hanno partecipato al progetto Erasmus.**

Select \*

from STUDENTE

where Erasmus is not NULL;

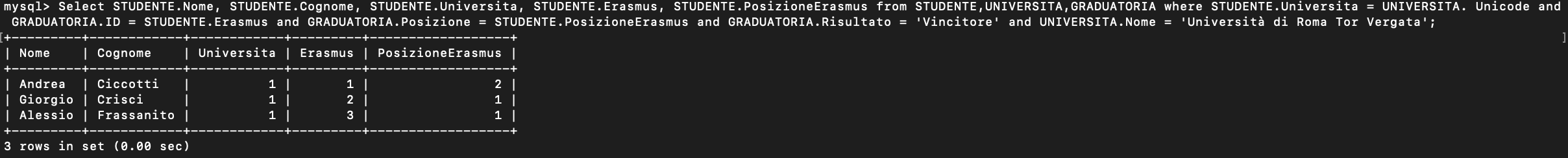


1. **Indicare quali sono gli studenti dell'Università di Roma Tor Vergata ad aver partecipato al progetto Erasmus risultandone vincitori.**

Select STUDENTE.Nome, STUDENTE.Cognome, STUDENTE.Universita, STUDENTE.Erasmus, STUDENTE.PosizioneErasmus

from STUDENTE,UNIVERSITA,GRADUATORIA

where STUDENTE.Universita = UNIVERSITA. Unicode and GRADUATORIA.ID = STUDENTE.Erasmus and GRADUATORIA.Posizione = STUDENTE.PosizioneErasmus and GRADUATORIA.Risultato = 'Vincitore' and UNIVERSITA.Nome = 'Università di Roma Tor Vergata';

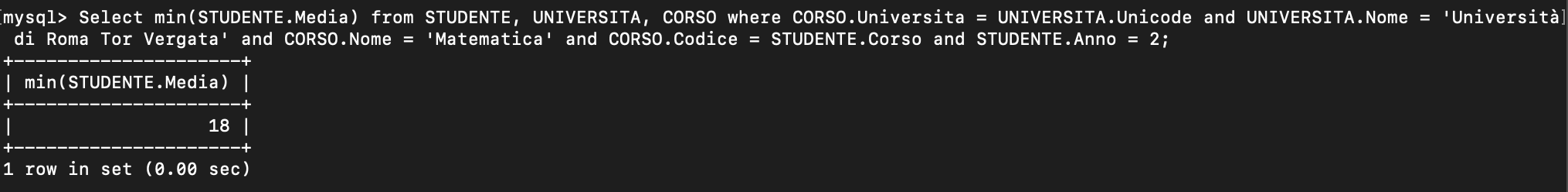


1. **Selezionare la media dei voti minima tra gli studenti iscritti al secondo anno del corso di Matematica dell'Università di Roma Tor Vergata.**

Select min(STUDENTE.Media)

from STUDENTE, UNIVERSITA, CORSO

where CORSO.Universita = UNIVERSITA.Unicode and UNIVERSITA.Nome = 'Università di Roma Tor Vergata' and CORSO.Nome = 'Matematica' and CORSO.Codice = STUDENTE.Corso and STUDENTE.Anno = 2;

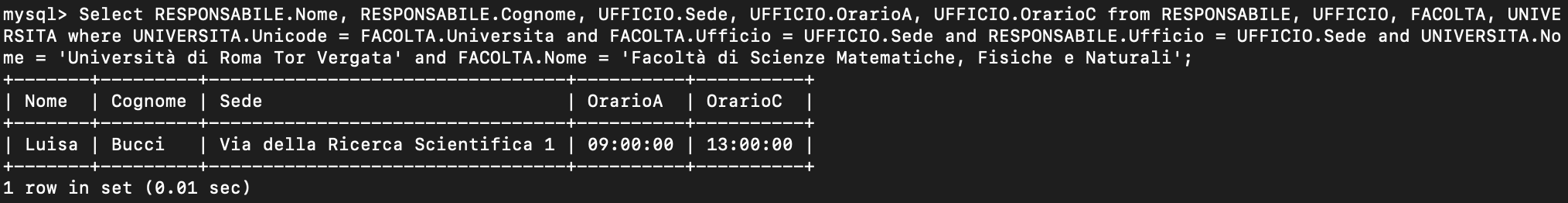


1. **Selezionare il responsabile dell'ufficio Erasmus della Facolta di Scienze Matematiche, Fisiche e Naturali dell'università di Roma Tor Vergata con il relativo orario di apertura dell'ufficio.**

Select RESPONSABILE.Nome, RESPONSABILE.Cognome, UFFICIO.Sede, UFFICIO.OrarioA, UFFICIO.OrarioC

from RESPONSABILE, UFFICIO, FACOLTA, UNIVERSITA

where UNIVERSITA.Unicode = FACOLTA.Universita and FACOLTA.Ufficio = UFFICIO.Sede and RESPONSABILE.Ufficio = UFFICIO.Sede and UNIVERSITA.Nome = 'Università di Roma Tor Vergata' and FACOLTA.Nome = 'Facoltà di Scienze Matematiche, Fisiche e Naturali';

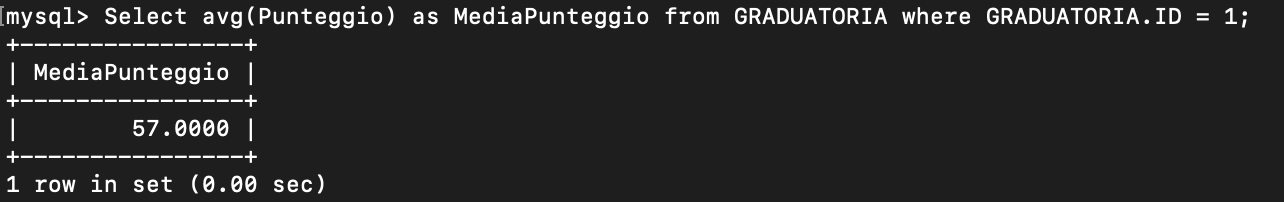


1. **Selezionare la media dei Punteggi della Graduatoria con ID = 1.**

Select avg(Punteggio) as MediaPunteggio

from GRADUATORIA

where GRADUATORIA.ID = 1;

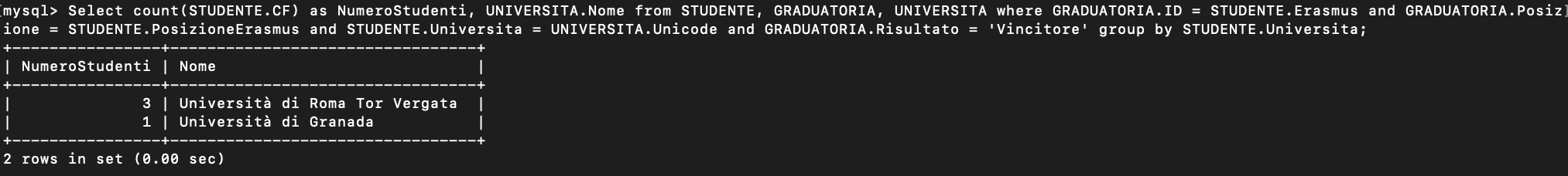


1. **Il numero di Studenti che hanno partecipato all'Erasmus per ogni Università.**

Select count(STUDENTE.CF) as NumeroStudenti, UNIVERSITA.Nome

from STUDENTE, GRADUATORIA, UNIVERSITA

where GRADUATORIA.ID = STUDENTE.Erasmus and GRADUATORIA.Posizione = STUDENTE.PosizioneErasmus and STUDENTE.Universita = UNIVERSITA.Unicode and GRADUATORIA.Risultato = 'Vincitore' group by STUDENTE.Universita;

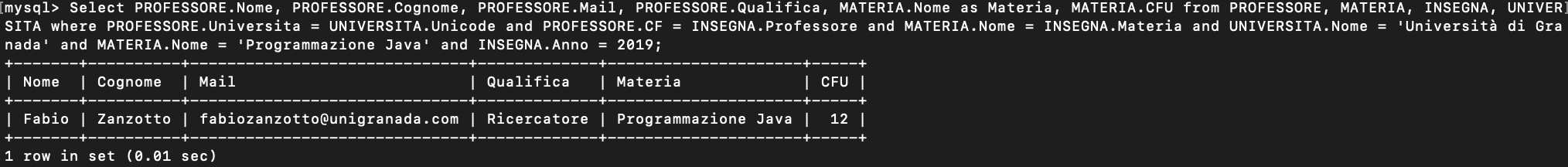


1. **Indicare il professore con la relativa mail e qualifica che insegna la materia 'Programmazione Java' all'università di Granada nel 2019 indicando quanti CFU ha la materia.**

Select PROFESSORE.Nome, PROFESSORE.Cognome, PROFESSORE.Mail, PROFESSORE.Qualifica, MATERIA.Nome as Materia, MATERIA.CFU

from PROFESSORE, MATERIA, INSEGNA, UNIVERSITA

where PROFESSORE.Universita = UNIVERSITA.Unicode and PROFESSORE.CF = INSEGNA.Professore and MATERIA.Nome = INSEGNA.Materia and UNIVERSITA.Nome = 'Università di Granada' and MATERIA.Nome = 'Programmazione Java' and INSEGNA.Anno = 2019;

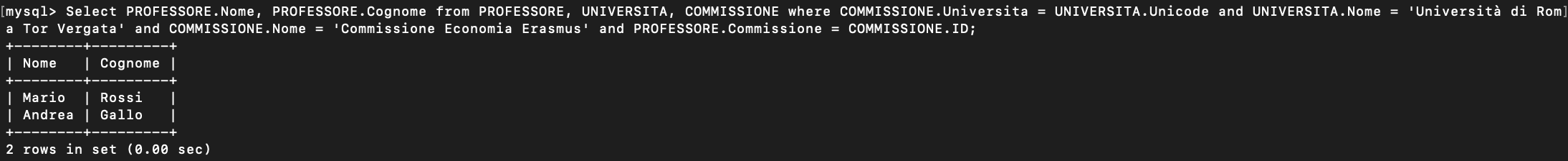


1. **Indicare quali professori hanno fatto parte della Commissione Economia Erasmus dell'Università di Roma Tor Vergata**

Select PROFESSORE.Nome, PROFESSORE.Cognome

from PROFESSORE, UNIVERSITA, COMMISSIONE

where COMMISSIONE.Universita = UNIVERSITA.Unicode and UNIVERSITA.Nome = 'Università di Roma Tor Vergata' and COMMISSIONE.Nome = 'Commissione Economia Erasmus' and PROFESSORE.Commissione = COMMISSIONE.ID;

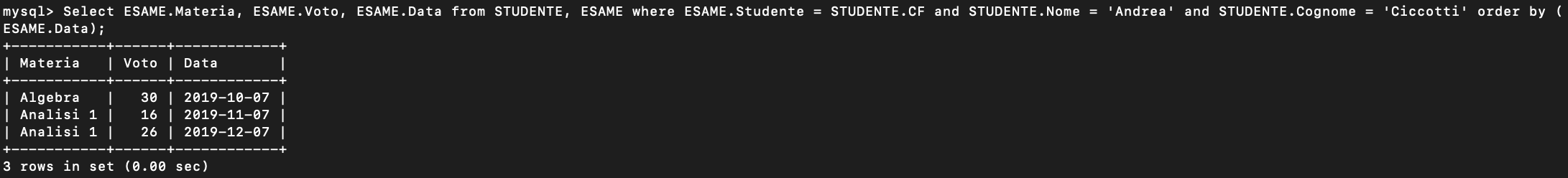


1. **Indicare quali esami con voto e data ha sostenuto lo studente Andrea Ciccotti in Erasmus ordinandoli per data**

Select ESAME.Materia, ESAME.Voto, ESAME.Data

from STUDENTE, ESAME

where ESAME.Studente = STUDENTE.CF and STUDENTE.Nome = 'Andrea' and STUDENTE.Cognome = 'Ciccotti' order by (ESAME.Data);

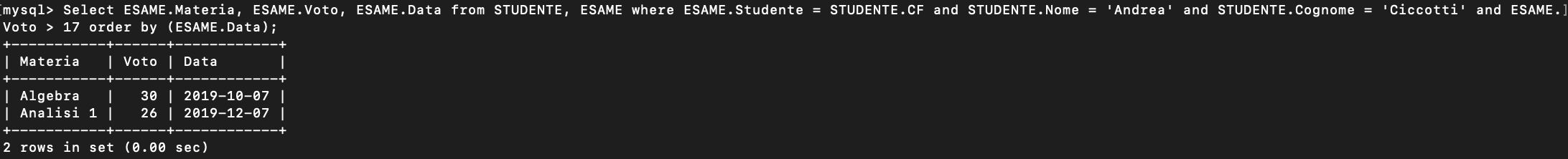


1. **Come la precedente solo che questa volta si vogliono indicare solo gli esami superati**

Select ESAME.Materia, ESAME.Voto, ESAME.Data

from STUDENTE, ESAME

where ESAME.Studente = STUDENTE.CF and STUDENTE.Nome = 'Andrea' and STUDENTE.Cognome = 'Ciccotti' and ESAME.Voto > 17 order by (ESAME.Data);

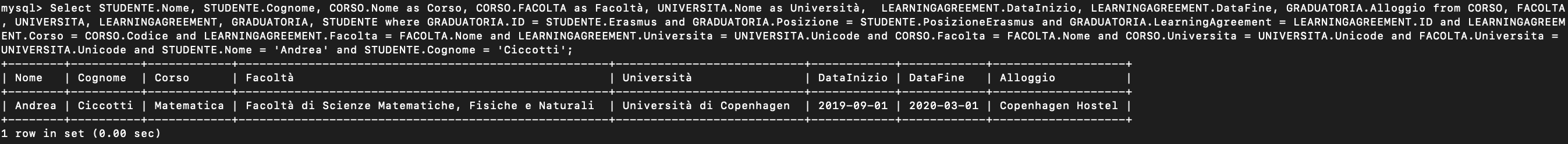


1. **Indicare dove è andato, in che periodo e in quale alloggio ha soggiornato Andrea Ciccotti in Erasmus**

Select STUDENTE.Nome, STUDENTE.Cognome, CORSO.Nome as Corso, CORSO.FACOLTA as Facoltà, UNIVERSITA.Nome as Università, LEARNINGAGREEMENT.DataInizio, LEARNINGAGREEMENT.DataFine, GRADUATORIA.Alloggio

from CORSO, FACOLTA, UNIVERSITA, LEARNINGAGREEMENT, GRADUATORIA, STUDENTE

where GRADUATORIA.ID = STUDENTE.Erasmus and GRADUATORIA.Posizione = STUDENTE.PosizioneErasmus and GRADUATORIA.LearningAgreement = LEARNINGAGREEMENT.ID and LEARNINGAGREEMENT.Corso = CORSO.Codice and LEARNINGAGREEMENT.Facolta = FACOLTA.Nome and LEARNINGAGREEMENT.Universita = UNIVERSITA.Unicode and CORSO.Facolta = FACOLTA.Nome and CORSO.Universita = UNIVERSITA.Unicode and FACOLTA.Universita = UNIVERSITA.Unicode and STUDENTE.Nome = 'Andrea' and STUDENTE.Cognome = 'Ciccotti';

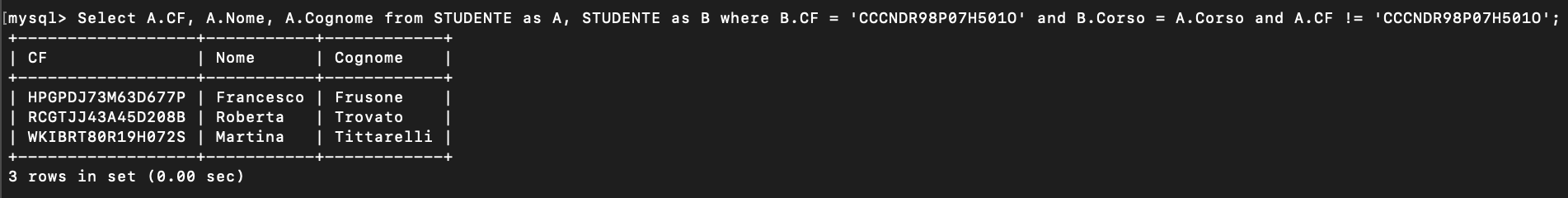


1. **Selezionare tutti gli studenti che sono iscritti allo stesso corso dello Studente con CF = ' CCCNDR98P07H501O'**

Select A.CF, A.Nome, A.Cognome

from STUDENTE as A, STUDENTE as B

where B.CF = 'CCCNDR98P07H501O' and B.Corso = A.Corso and A.CF != 'CCCNDR98P07H501O';

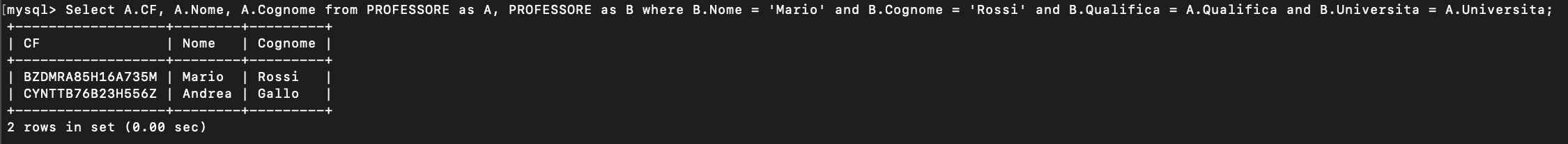


1. **Selezionare tutti i professori che hanno la stessa qualifica di Mario Rossi e lavorano per la stessa università.**

Select A.CF, A.Nome, A.Cognome

from PROFESSORE as A, PROFESSORE as B

where B.Nome = 'Mario' and B.Cognome = 'Rossi' and B.Qualifica = A.Qualifica and B.Universita = A.Universita;



1. **Selezionare la media degli affitti degli alloggi per ogni nazione**

Select avg(Affitto) as MediaAffitto, Nazione

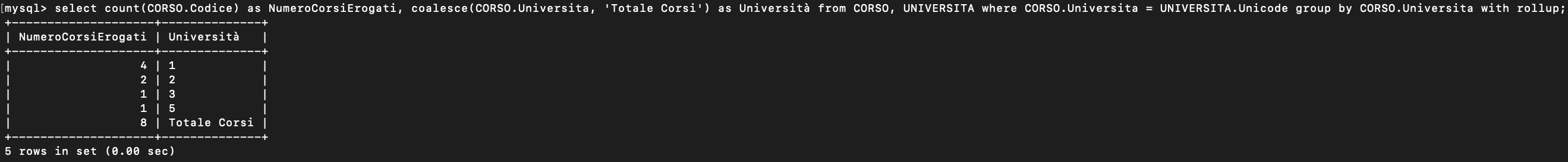
from ALLOGGIO group by Nazione;



1. **Contare i corsi per ogni università con il totale finale**

Select count(CORSO.Codice) as NumeroCorsiErogati, coalesce(CORSO.Universita, 'Totale Corsi') as Università from CORSO, UNIVERSITA

where CORSO.Universita = UNIVERSITA.Unicode group by CORSO.Universita with rollup;

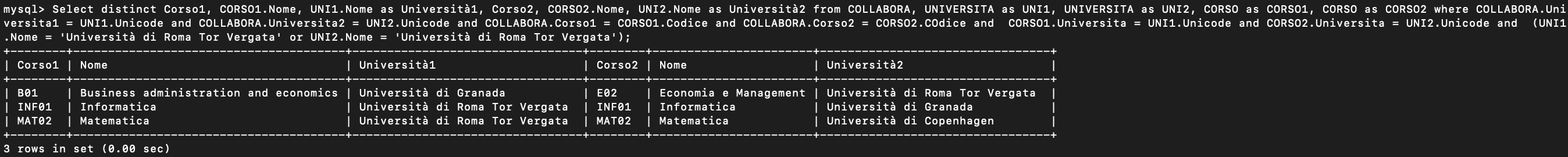


1. **Selezionare le collaborazioni Erasmus che l'Università di Roma Tor Vergata ha fatto sia in entrata che in uscita con altre Università prendendo in considerazione una sola volta le collaborazioni uguali.**

Select distinct Corso1, CORSO1.Nome, UNI1.Nome as Università1, Corso2, CORSO2.Nome, UNI2.Nome as Università2

from COLLABORA, UNIVERSITA as UNI1, UNIVERSITA as UNI2, CORSO as CORSO1, CORSO as CORSO2

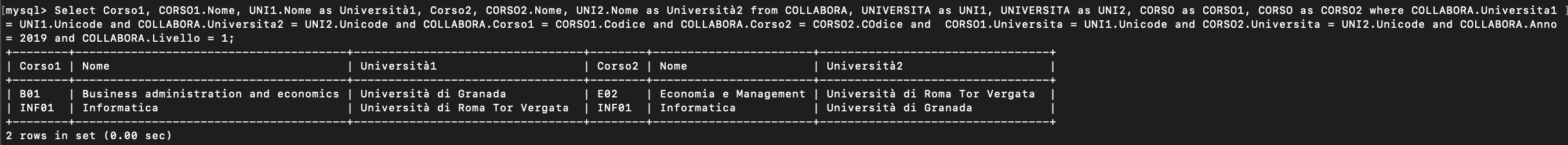
where COLLABORA.Universita1 = UNI1.Unicode and COLLABORA.Universita2 = UNI2.Unicode and COLLABORA.Corso1 = CORSO1.Codice and COLLABORA.Corso2 = CORSO2.COdice and CORSO1.Universita = UNI1.Unicode and CORSO2.Universita = UNI2.Unicode and (UNI1.Nome = 'Università di Roma Tor Vergata' or UNI2.Nome = 'Università di Roma Tor Vergata');



1. **Selezionare le collaborazioni Erasmus di livello 1 che ci sono state tra le varie università nel 2019**

Select Corso1, CORSO1.Nome, UNI1.Nome as Università1, Corso2, CORSO2.Nome, UNI2.Nome as Università2 from COLLABORA, UNIVERSITA as UNI1, UNIVERSITA as UNI2, CORSO as CORSO1, CORSO as CORSO2

where COLLABORA.Universita1 = UNI1.Unicode and COLLABORA.Universita2 = UNI2.Unicode and COLLABORA.Corso1 = CORSO1.Codice and COLLABORA.Corso2 = CORSO2.COdice and CORSO1.Universita = UNI1.Unicode and CORSO2.Universita = UNI2.Unicode and COLLABORA.Anno = 2019 and COLLABORA.Livello = 1;

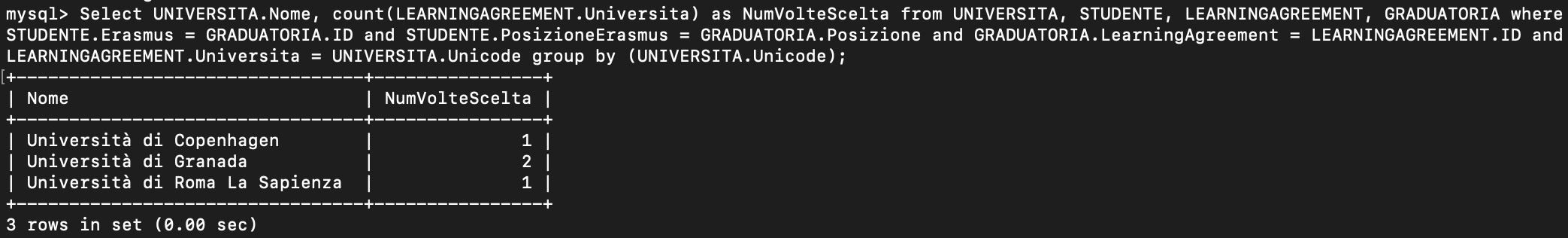


1. **Selezionare per ogni Università quante volte è stata scelta dagli studenti Erasmus**

Select UNIVERSITA.Nome, count(LEARNINGAGREEMENT.Universita) as NumVolteScelta

from UNIVERSITA, STUDENTE, LEARNINGAGREEMENT, GRADUATORIA

where STUDENTE.Erasmus = GRADUATORIA.ID and STUDENTE.PosizioneErasmus = GRADUATORIA.Posizione and GRADUATORIA.LearningAgreement = LEARNINGAGREEMENT.ID and LEARNINGAGREEMENT.Universita = UNIVERSITA.Unicode group by (UNIVERSITA.Unicode);

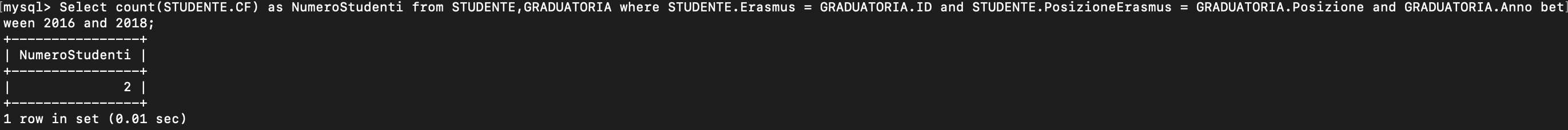


1. **Il numero di studenti che hanno fatto domanda per partecipare al bando Erasmus tra il 2016 e il 2018**

Select count(STUDENTE.CF) as NumeroStudenti

from STUDENTE,GRADUATORIA

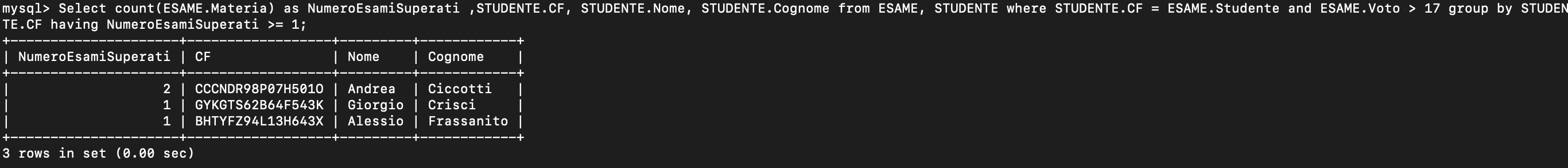
where STUDENTE.Erasmus = GRADUATORIA.ID and STUDENTE.PosizioneErasmus = GRADUATORIA.Posizione and GRADUATORIA.Anno between 2016 and 2018;



1. **Indicare quali sono gli studenti ad aver superato almeno 1 esame in Erasmus**

Select count(ESAME.Materia) as NumeroEsamiSuperati , STUDENTE.CF, STUDENTE.Nome, STUDENTE.Cognome from ESAME, STUDENTE

where STUDENTE.CF = ESAME.Studente and ESAME.Voto > 17 group by STUDENTE.CF having NumeroEsamiSuperati >= 1;



**Algebra Relazionale**

**Query 1: Selezionare le Nazioni con contributo mensile > 250;**

π[NAZIONE.Nome, NAZIONE.ContributoMensile](σ[ContributoMensile > 250] (ρ[ContributoMensile ⇽ NAZIONE.Contributo] (NAZIONE))

**Query 10: Indicare quali professori hanno fatto parte della Commissione Economia Erasmus dell’Università di Roma Tor Vergata**

π[PROFESSORE.Nome, PROFESSORE.Cognome](PROFESSORE |X|[Commissione = ID] σ[COMMISSIONE.Nome = ‘Commissione Erasmus Scienze’](COMMISSIONE) |X| [Università = Unicode]σ[UNIVERSITA.Nome = ‘Università di Roma Tor Vergata’](UNIVERSITA))

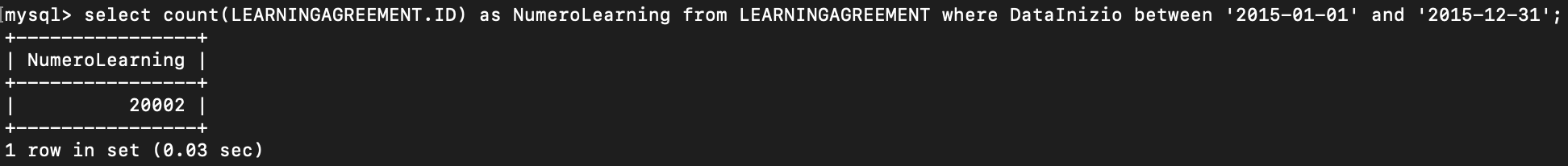
**Ottimizzazione**

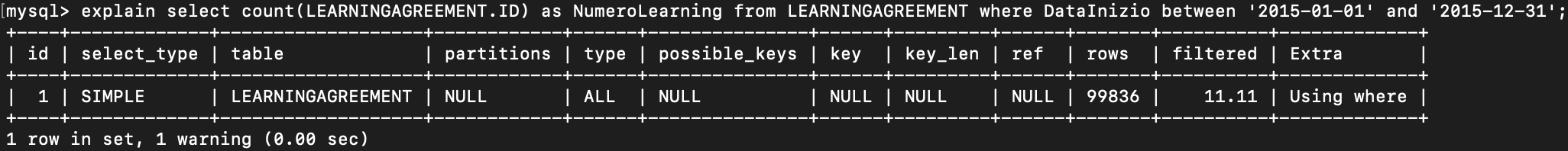
Vediamo orala parte di ottimizzazione del database. Per il testing sono stati aggiunti in maniera casuale 100000 record ad ogni entità, fatta eccezione dell’entità Nazione che contiene circa 200 record. I dati sono stati generati random tramite il software **Spawner Data Generator**.

**INDICI**

Uno dei principali metodi di ottimizzazione è l’utilizzo di indici, che consentono un accesso efficiente ai dati. L’utilizzo di un indice è efficiente quando la quantità di righe ritornata è in media attorno al 10%. Questo è dovuto al fatto che l’accesso casuale è molto più lento del Full-Table Scan a causa del seek time.

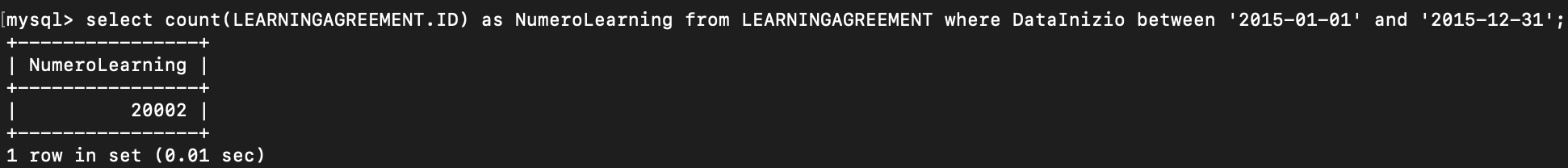
Per esempio potrebbe essere utile creare un indice sulla tabella LEARNINGAGREEMENT colonna ‘DataInizio’ in modo tale da accedere in maniera più veloce alle varie collaborazioni Erasmus che iniziano in un certo periodo.

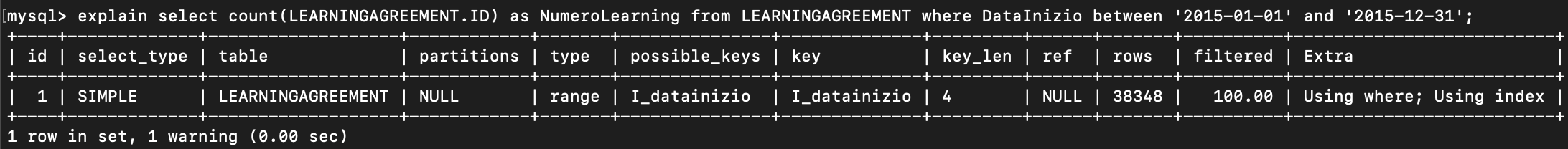


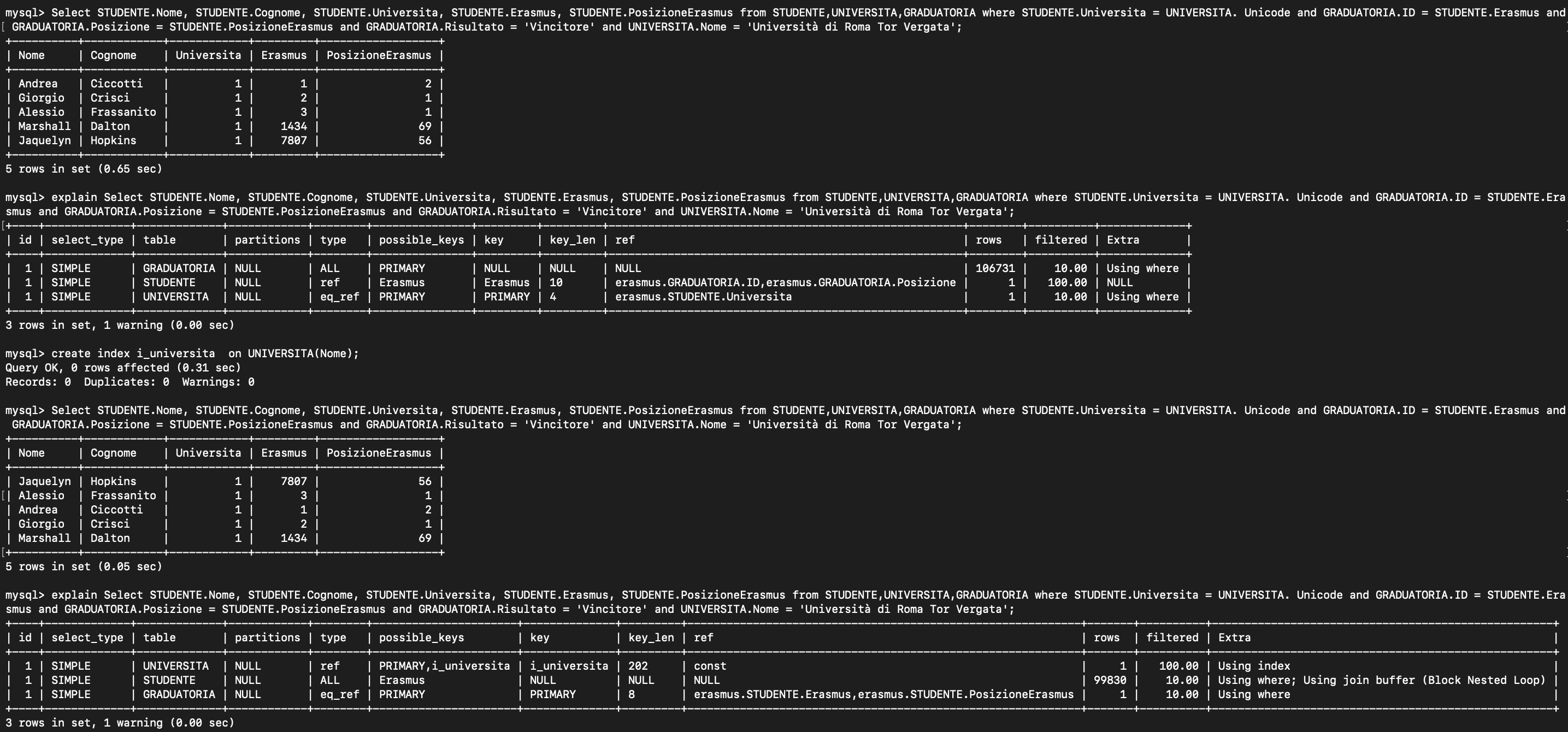


Creando l’indice possiamo vedere come il tempo cali drasticamente e come il numero di rows sia passato da 99836 a 38348, quasi tre volte di meno.

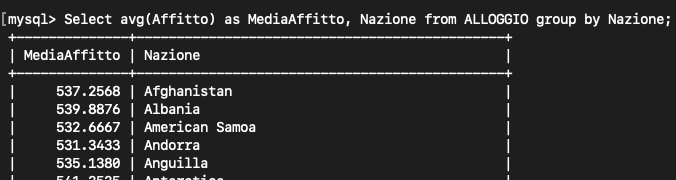
CREATE INDEX I\_dataInizio on LEARNINGAGREEMENT(DataInizio);



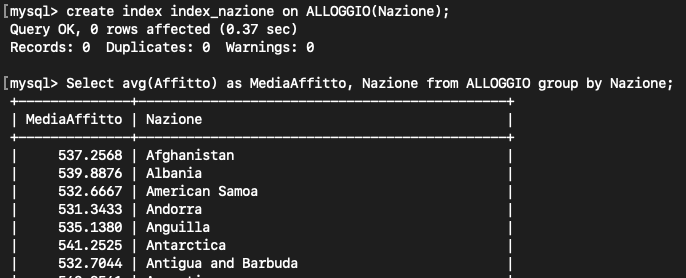


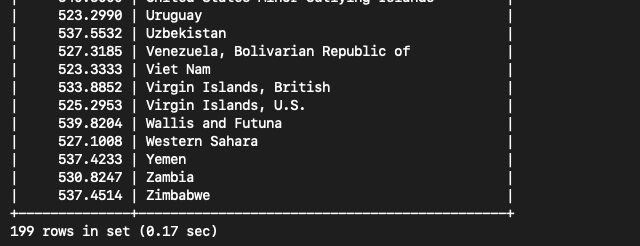
Analogamente possiamo ottimizzare la **query 4** scritta sopra:

Analogamente è possibile ottimizzare la **query 16** scritta sopra:









**TRIGGER**

Un trigger è un tipo di Stored Procedure eseguito automaticamente all’avvenire di un determinato evento prefissato nel codice del trigger stesso.

**1.Trigger per accertarsi che DataInizio si antecedente a DataFine sulla tabella LEARNINGAGREEMENT**

DELIMITER //

CREATE TRIGGER TR\_LEARNINGAGREEMENT\_Data

BEFORE INSERT ON LEARNINGAGREEMENT

FOR EACH ROW

BEGIN

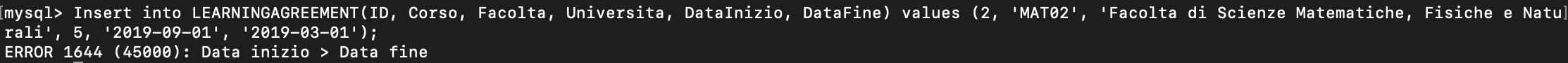
IF(new.DataInizio > new.DataFine) then SIGNAL SQLSTATE ‘45000’

SET MESSAGE\_TEXT = ‘Data inizio > Data fine’;

END IF;

END; //

DELIMITER ;



**2.Trigger per controllare che il voto di un esame sia compreso tra 0 e 31**

DELIMITER //

CREATE TRIGGER TR\_ESAME\_Voto

BEFORE INSERT ON ESAME

FOR EACH ROW

BEGIN

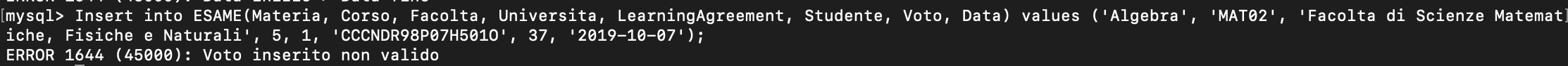
IF(new.Voto < 0 OR new.Voto > 31) then SIGNAL SQLSTATE ‘45000’

SET MESSAGE\_TEXT = ‘Voto inserito non valido’;

END IF;

END; //

DELIMITER ;



**3.Trigger per controllare che un professore faccia parte solo delle commissioni della propria università**

DELIMITER //

CREATE TRIGGER TR\_PROFESSORE\_Commissione

BEFORE INSERT ON PROFESSORE

FOR EACH ROW

BEGIN

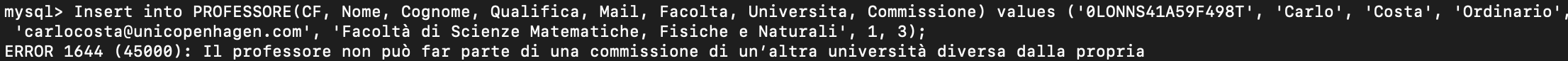
IF(new.Universita != (select COMMISSIONE.Universita from COMMISSIONE where COMMISSIONE.ID = new.Commissione)) then SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Il professore non può far parte di una commissione di un’altra università diversa dalla propria';

END IF;

END; //

DELIMITER ;

****

**4.Trigger per controllare che uno studente scelga come destinazione Erasmus un Corso di una Università che collabora con il corso della propria Università**

DELIMITER //

CREATE TRIGGER TR\_STUDENTE\_Erasmus

BEFORE INSERT ON STUDENTE

FOR EACH ROW

BEGIN

DECLARE LearningAgreementScelto int;

DECLARE CorsoScelto varchar(50);

DECLARE FacoltaScelta varchar(50);

DECLARE UniScelta int;

SET LearningAgreementScelto = (select GRADUATORIA.LearningAgreement from GRADUATORIA where GRADUATORIA.ID = new.Erasmus and GRADUATORIA.Posizione = new.PosizioneErasmus);

SET CorsoScelto = (select LEARNINGAGREEMENT.Corso from LEARNINGAGREEMENT where LEARNINGAGREEMENT.ID = LearningAgreementScelto);

SET FacoltaScelta = (select LEARNINGAGREEMENT.Facolta from LEARNINGAGREEMENT where LEARNINGAGREEMENT.ID = LearningAgreementScelto);

SET UniScelta = (select LEARNINGAGREEMENT.Universita from LEARNINGAGREEMENT where LEARNINGAGREEMENT.ID = LearningAgreementSCelto);

IF( not exists (select \* from COLLABORA where (Corso1 = CorsoScelto and Facolta1 = FacoltaScelta and Universita1 = UniScelta and Corso2 = new.Corso and Facolta2 = new.Facolta and Universita2 = new.Universita) OR (Corso2 = CorsoScelto and Facolta2 = FacoltaScelta and Universita2 = UniScelta and Corso1 = new.Corso and Facolta1 = new.Facolta and Universita1 = new.Universita))) then SIGNAL SQLSTATE '45000'

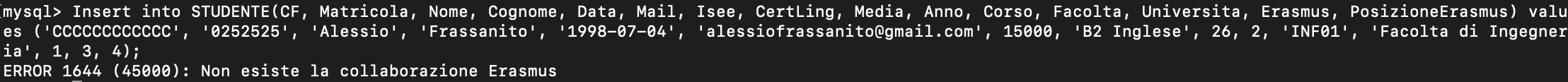
Set MESSAGE\_TEXT = 'Non esiste la collaborazione Erasmus';

End IF;

END;

//

DELIMITER ;



**5.Trigger per controllare che uno studente partecipi alla graduatoria Erasmus stilata dalla sua facoltà dell’università dove è iscritto**

DELIMITER //

CREATE TRIGGER TR\_STUDENTE\_Graduatoria

BEFORE INSERT ON STUDENTE

FOR EACH ROW

BEGIN

DECLARE CommissioneScelta int;

DECLARE UniCommissioneScelta int;

DECLARE FacoltaCommissioneScelta varchar(50);

SET CommissioneScelta = (select GRADUATORIA.Commissione from GRADUATORIA where new.Erasmus = GRADUATORIA.ID and new.PosizioneErasmus = GRADUATORIA.Posizione);

SET UniCommissioneScelta = (select COMMISSIONE.Universita from COMMISSIONE where COMMISSIONE.ID = CommissioneScelta);

SET FacoltaCommissioneScelta = (select COMMISSIONE.Facolta from COMMISSIONE where COMMISSIONE.ID = CommissioneScelta);

IF((new.Universita != UniCommissioneScelta) OR (new.Facolta != FacoltaCommissioneScelta)) then SIGNAL SQLSTATE '45000'

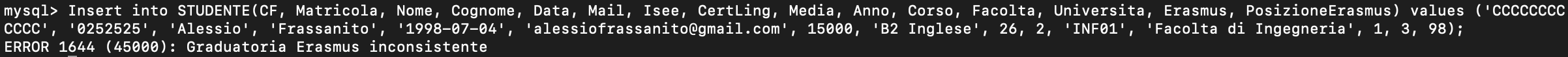
Set MESSAGE\_TEXT = 'Graduatoria Erasmus inconsistente';

End IF;

END;

//

DELIMITER ;



**6.Trigger per controllare che la data d’esame sia compresa tra la data d’inizio e di fine del Learning Agreement**

DELIMITER //

CREATE TRIGGER TR\_ESAME\_Data

BEFORE INSERT ON ESAME

FOR EACH ROW

BEGIN

DECLARE DataI Date;

DECLARE DataF Date;

SET DataI = (select DataInizio from LEARNINGAGREEMENT where new.LearningAgreement = LEARNINGAGREEMENT.ID);

SET DataF = (select DataFine from LEARNINGAGREEMENT where new.LearningAgreement = LEARNINGAGREEMENT.ID);

IF(new.Data not between DataI and DataF) then SIGNAL SQLSTATE ‘45000’

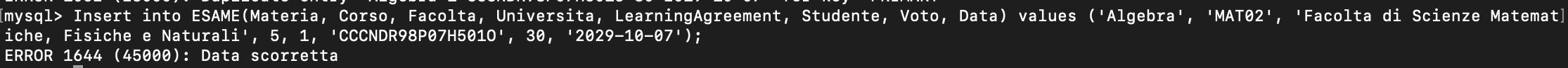
SET MESSAGE\_TEXT = ‘Data scorretta’;

End IF;

END;

//

DELIMITER ;



**7.Trigger per controllare che se lo studente non è vincitore allora il LearningAgreement e l’Alloggio devono essere impostato a NULL in quanto non devono essere compilati/scelti.**

DELIMITER //

CREATE TRIGGER TR\_GRADUATORIA\_LearningAgreement

BEFORE INSERT ON GRADUATORIA

FOR EACH ROW

BEGIN

IF(new.Risultato != “Vincitore”) THEN

SET new.LearningAgreement = NULL;

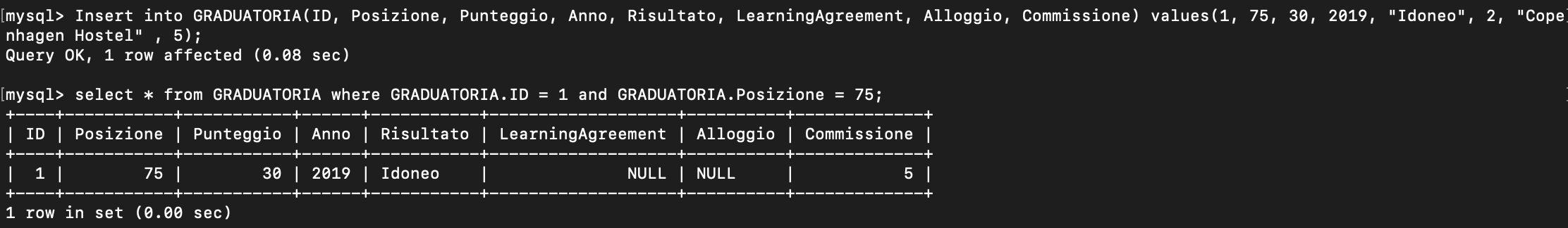
SET new.Alloggio = NULL;

END IF;

END;

//

DELIMITER ;



**STORED PROCEDURE E CURSORI**

Le procedure permettono di associare un nome a un’istruzione SQL, con la possibilità di specificare dei parametri da utilizzare per lo scambio di informazioni con la procedura. I vantaggi sono un aumento della comprensibilità del programma, una più facile manutenibilità e, la possibilità di ottenere in diversi casi un sensibile incremento delle prestazioni.

**1.Stored procedure che dati in input nome e cognome restituisce il CF del primo studente corrispondente.**

DROP PROCEDURE IF EXISTS trovaStudente;

DELIMITER //

CREATE PROCEDURE trovaStudente(IN nome VARCHAR(50), cognome VARCHAR(50), OUT cf VARCHAR(16))

BEGIN

   DECLARE appoggio VARCHAR(50);

   DECLARE cercaStudente CURSOR FOR SELECT STUDENTE.CF FROM STUDENTE WHERE STUDENTE.Nome = nome and STUDENTE.Cognome = cognome;

   OPEN cercaStudente;

   FETCH cercaStudente INTO appoggio;

   CLOSE cercaStudente;

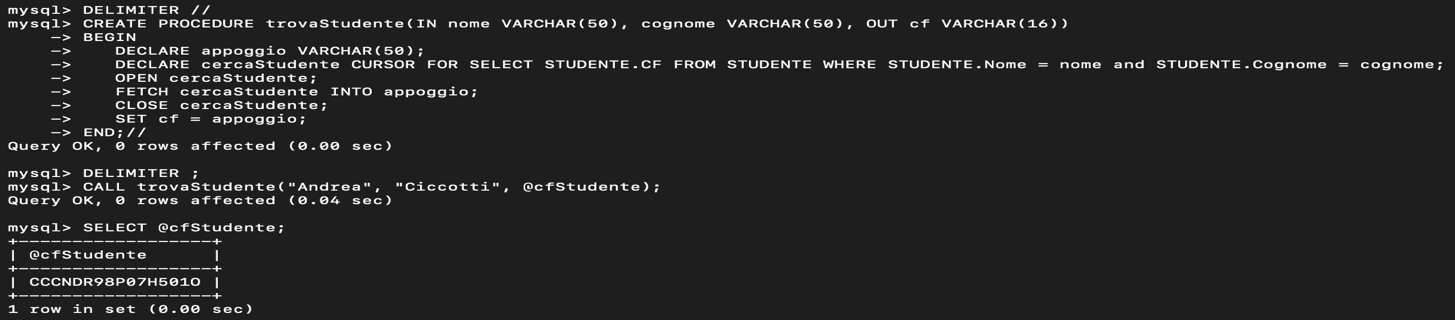
   SET cf = appoggio;

END;//

DELIMITER ;

CALL trovaStudente("Andrea", "Ciccotti", @cfStudente);

SELECT @cfStudente;



**2.Stored procedure che dato in input il CF di uno studente dice se lo studente è in corso o fuori corso.**

DROP PROCEDURE IF EXISTS corsoOFuoriCorso;

delimiter //

CREATE PROCEDURE corsoOFuoriCorso(IN CF VARCHAR(16), OUT InCorsoOFuoriCorso VARCHAR(50))

BEGIN

DECLARE corso varchar(50);

DECLARE tipologia varchar(50);

DECLARE anno int;

SELECT STUDENTE.Corso, CORSO.Tipologia, STUDENTE.Anno into corso, tipologia, anno FROM STUDENTE, CORSO WHERE STUDENTE.CF = CF and STUDENTE.Corso = CORSO.Codice and STUDENTE.Facolta = CORSO.Facolta and STUDENTE.Universita = CORSO.Universita;

if (tipologia = 'Corso di laurea triennale' and anno > 3) then

set InCorsoOFuoriCorso = 'Studente triennale fuori corso';

End if;

If (tipologia = 'Corso di laurea triennale' and anno < 4) then

Set InCorsoOFuoriCorso = 'Studente triennale in corso';

End if;

If (tipologia = 'Corso di laurea magistrale' and anno > 2) then

Set InCorsoOFuoriCorso = 'Studente magistrale fuori corso';

End if;

If (tipologia = 'Corso di laurea magistrale' and anno < 3) then

Set InCorsoOFuoriCorso = 'Studente magistrale in corso';

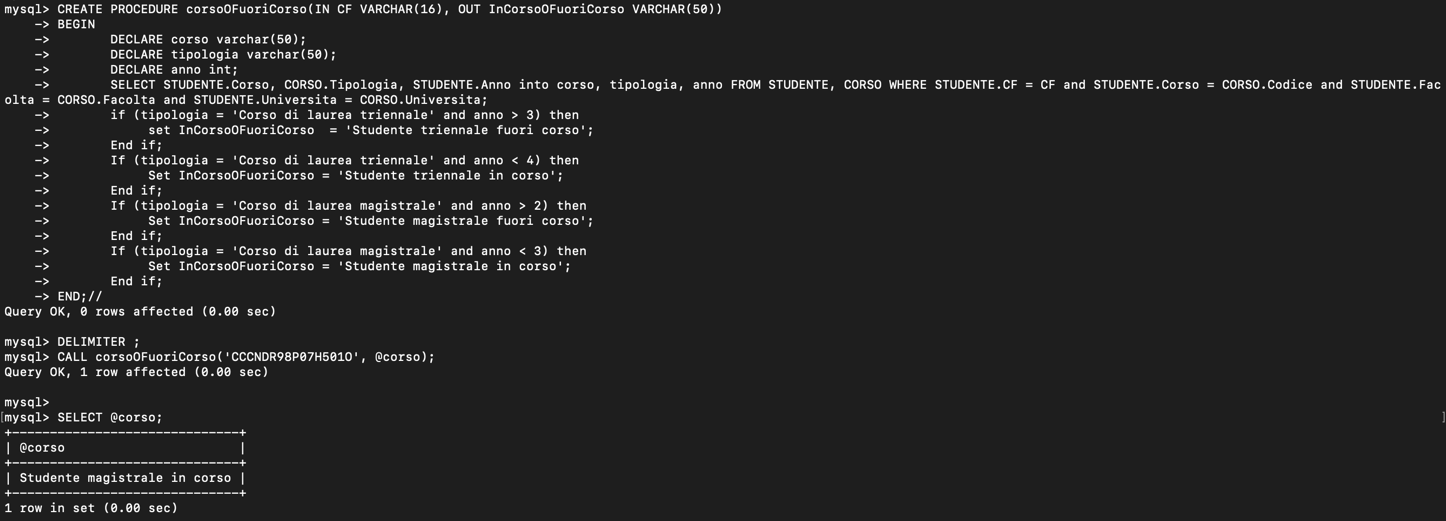
End if;

END;//

DELIMITER ;

CALL corsoOFuoriCorso('CCCNDR98P07H501O', @corso);

SELECT @corso;



**3.Stored procedure che dato un’universita restituisce tutte gli alloggi in quella determinata nazione dove si trova l’università**

DROP PROCEDURE IF EXISTS UniAlloggio;

DELIMITER //

CREATE PROCEDURE UniAlloggio(IN Universita int)

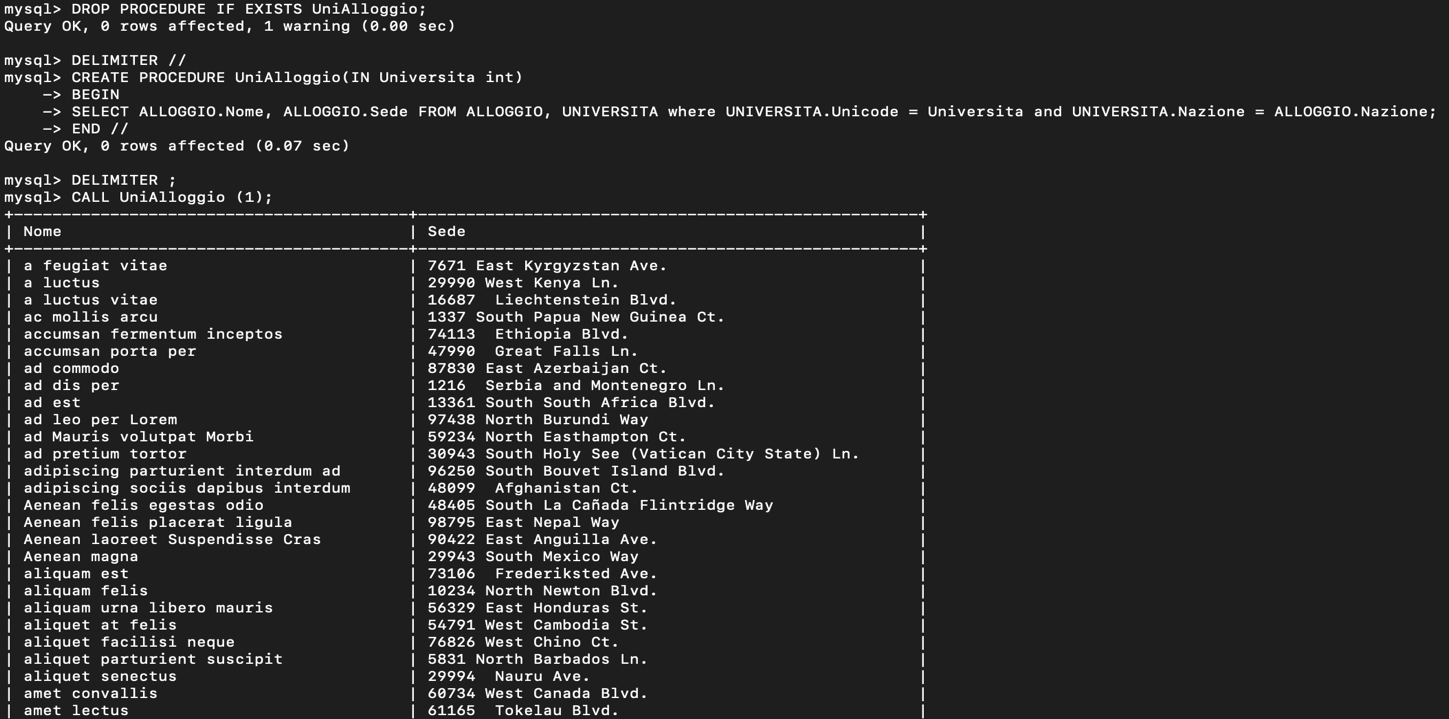
BEGIN

SELECT ALLOGGIO.Nome, ALLOGGIO.Sede FROM ALLOGGIO, UNIVERSITA where UNIVERSITA.Unicode = Universita and UNIVERSITA.Nazione = ALLOGGIO.Nazione;

END //

DELIMITER ;

CALL UniAlloggio (1);



**VISTE**

##### Una view è una tabella virtuale che consiste in una query memorizzata con un proprio nome. A seguire alcuni esempi di utilizzo di view nel caso di questo database.

**Vista per calcolare la media dei punteggi di ogni graduatoria**

CREATE VIEW MediaPunteggiGraduatoria(Graduatoria, MediaPunteggio) as

SELECT GRADUATORIA.ID, avg(GRADUATORIA.Punteggio) as MediaPunteggio

FROM GRADUATORIA group by GRADUATORIA.ID;

**Vista che contiene tutti gli studenti risultati ‘Idonei’ in graduatoria**

CREATE VIEW StudentiIdonei(CF, Nome, Cognome) as

SELECT STUDENTE.CF , STUDENTE.Nome, STUDENTE.Cognome

FROM STUDENTE, GRADUATORIA

WHERE STUDENTE.Erasmus = GRADUATORIA.ID and STUDENTE.PosizioneErasmus = GRADUATORIA.Posizione and GRADUATORIA.Risultato = ‘Idoneo’;

**Vista che contiene tutti i dati di uno studente tranne quelli più riservati per un discorso di privacy**

CREATE VIEW StudentePrivacy(Matricola, Universita, Erasmus, PosizioneErasmus) as

SELECT STUDENTE.Matricola, STUDENTE.Universita, STUDENTE.Erasmus, STUDENTE.PosizioneErasmus

FROM STUDENTE;

**MongoDB**

MongoDB è uno dei più diffusi database non relazionali, anche detti NoSQL, che utilizza uno stile a schema dinamico per la rappresentazione dei propri dati.

Tra i punti chiave di MongoDB troviamo: -Query ad hoc -Indicizzazione -Alta affidabilità -Sharding e bilanciamento dei dati – File storage -Aggregazione

**CREAZIONE DATABASE**

Il passo iniziale è quello della creazione del Database.

In mongo si utilizza la sintassi “use NOME\_DB”, che:

-Creerà un nuovo database nel caso uno con quel nome non sia già presente

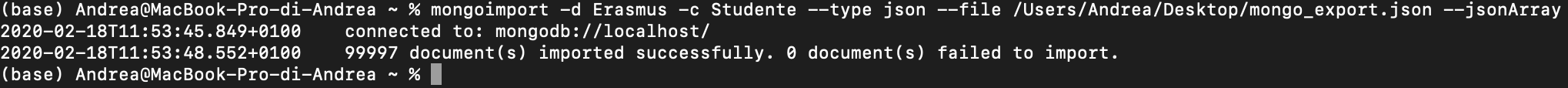
-Andrà ad utilizzare il database appena creato (o già esistente)



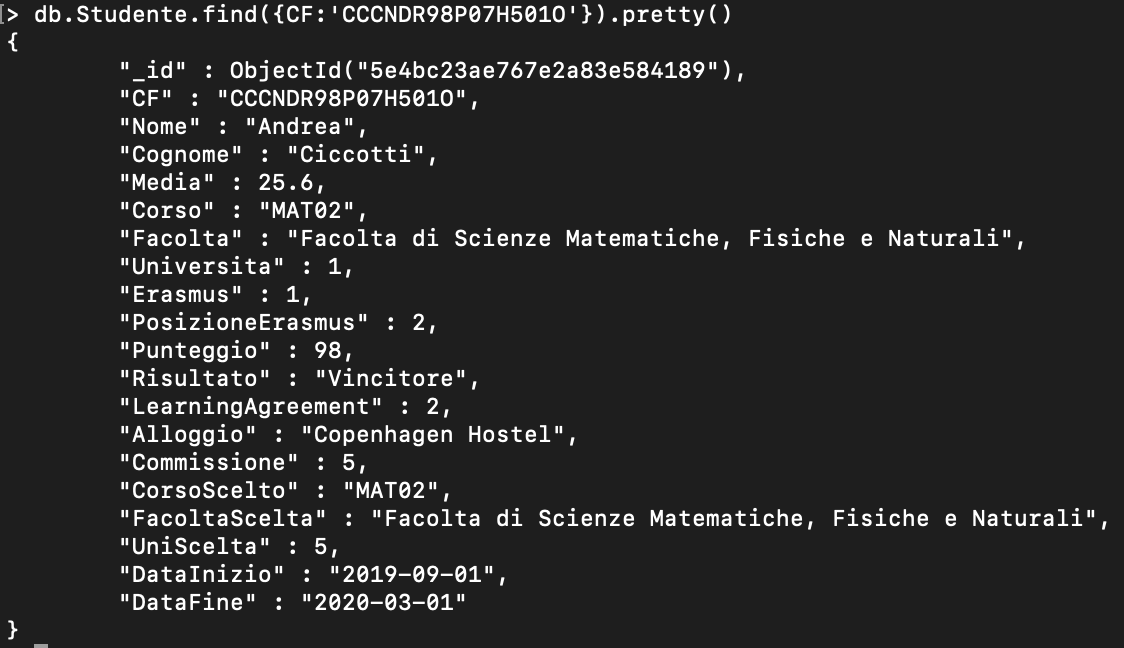
Per l’esportazione, inizio con la creazione di una **View** sui rami STUDENTE, GRADUATORIA e LEARNINGAGREEMENT

mysqle view Mongo\_Export as select STUDENTE.CF, STUDENTE.Nome, STUDENTE.Cognome, STUDENTE.Media, STUDENTE.Corso, STUDENTE.Facolta, STUDENTE.Universita, STUDENTE.Erasmus, STUDENTE.PosizioneErasmus, GRADUATORIA.Punteggio, GRADUATORIA.Risultato, GRADUATORIA.LearningAgreement, GRADUATORIA.Alloggio, GRADUATORIA.Commissione, LEARNINGAGREEMENT.Corso as CorsoScelto, LEARNINGAGREEMENT.Facolta as FacoltaScelta, LEARNINGAGREEMENT.Universita as UniScelta, LEARNINGAGREEMENT.DataInizio, LEARNINGAGREEMENT.DataFine from STUDENTE, GRADUATORIA, LEARNINGAGREEMENT where STUDENTE.Erasmus = GRADUATORIA.ID and STUDENTE.PosizioneErasmus = GRADUATORIA.Posizione and GRADUATORIA.LearningAgreement = LEARNINGAGREEMENT.ID;

Esporto successivamente l’intero contenuto della View in un file **json**, che chiamaremo Mongo\_Export.json, e procediamo a importarlo su MongoDB attraverso questo comando, ottenendo la **Collection** Studente



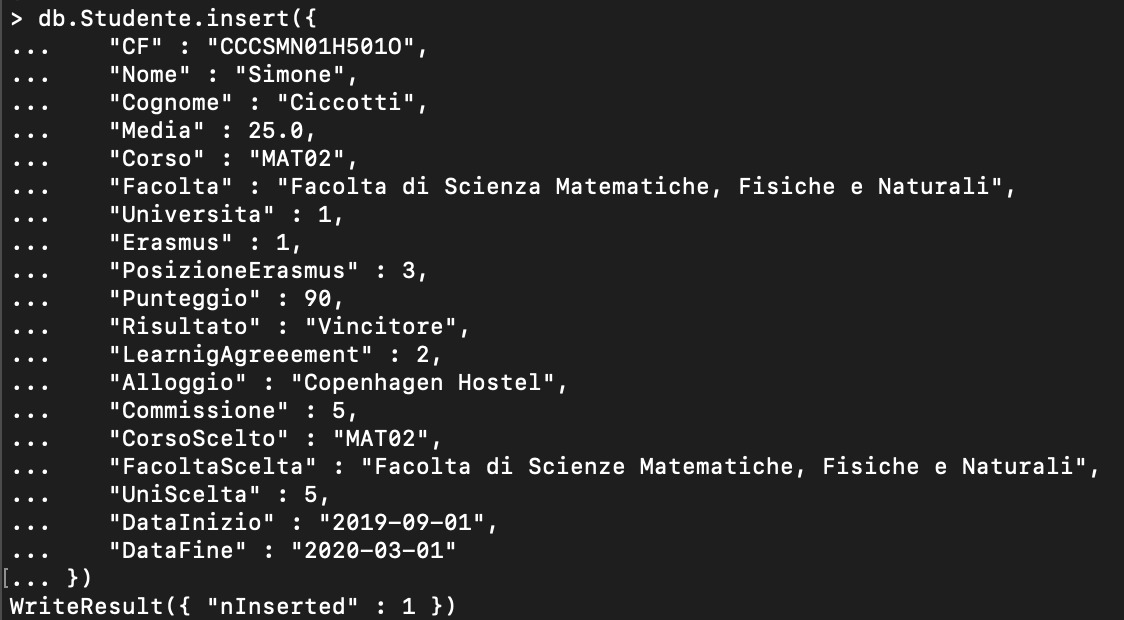
Effettuando una **Find** possiamo ottenere un estratto del contenuto presente nella collection.



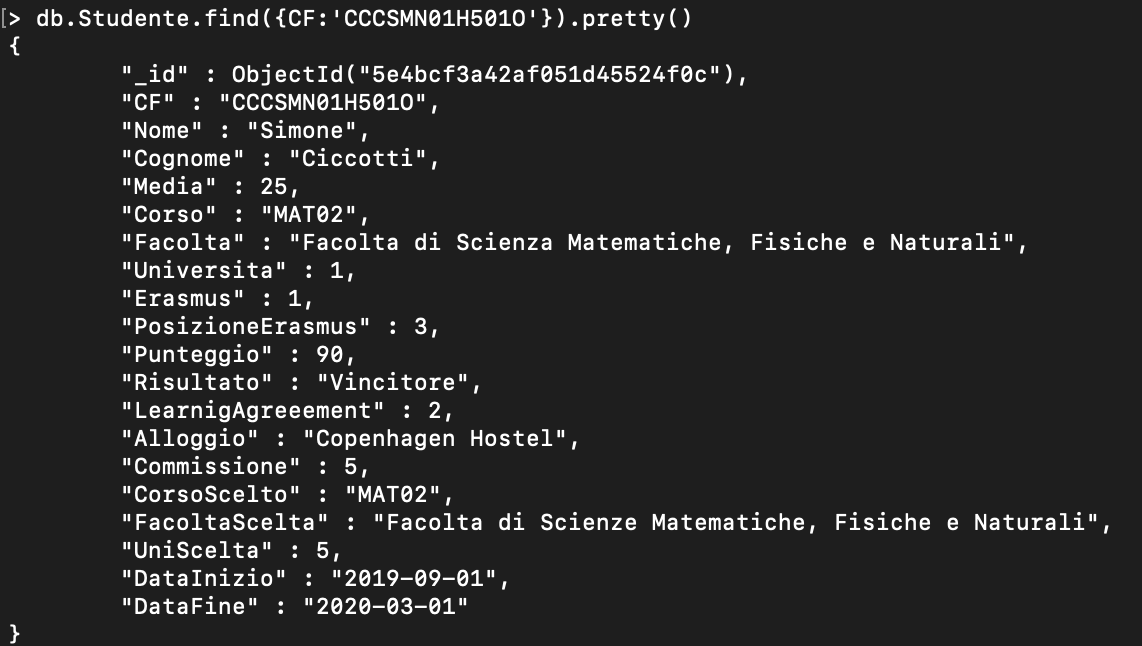
Ora che il ramo del database è importato, procediamo con l’analizzare le operazioni elementari in MongoDB.

**INSERT**

L’inserimento in MongoDB avviene tramite la sintassi db.collection.insert({Campo1:’Value1’, Campo2:’Value2’,….})

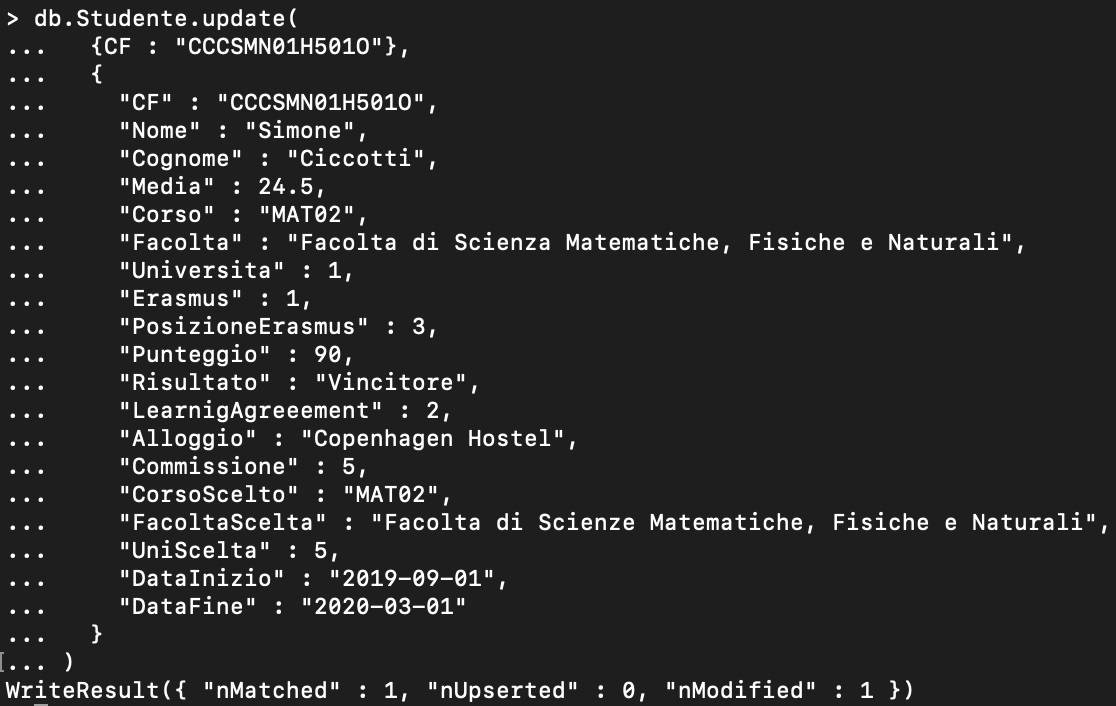


Eseguendo una find possiamo verificare il corretto inserimento:

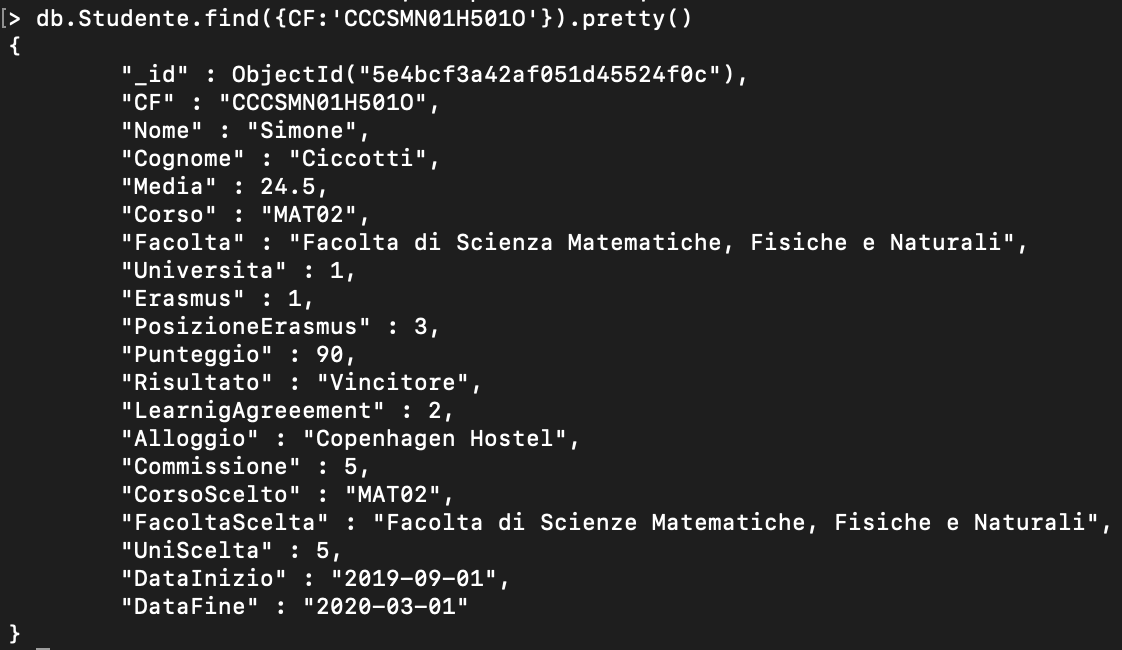


**UPDATE**

L’aggiornamento in MongoDB si ottiene attraverso il comando update che richiede due parametri: il primo per identificare il record da trovare e il secondo è invece la lista dei nuovi valori ai relativi campi del record. Nell’esempio aggiorniamo il campo Media.

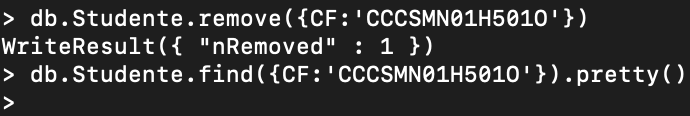


Verifichiamo ora che il record sia correttamente modificato tramite una find.



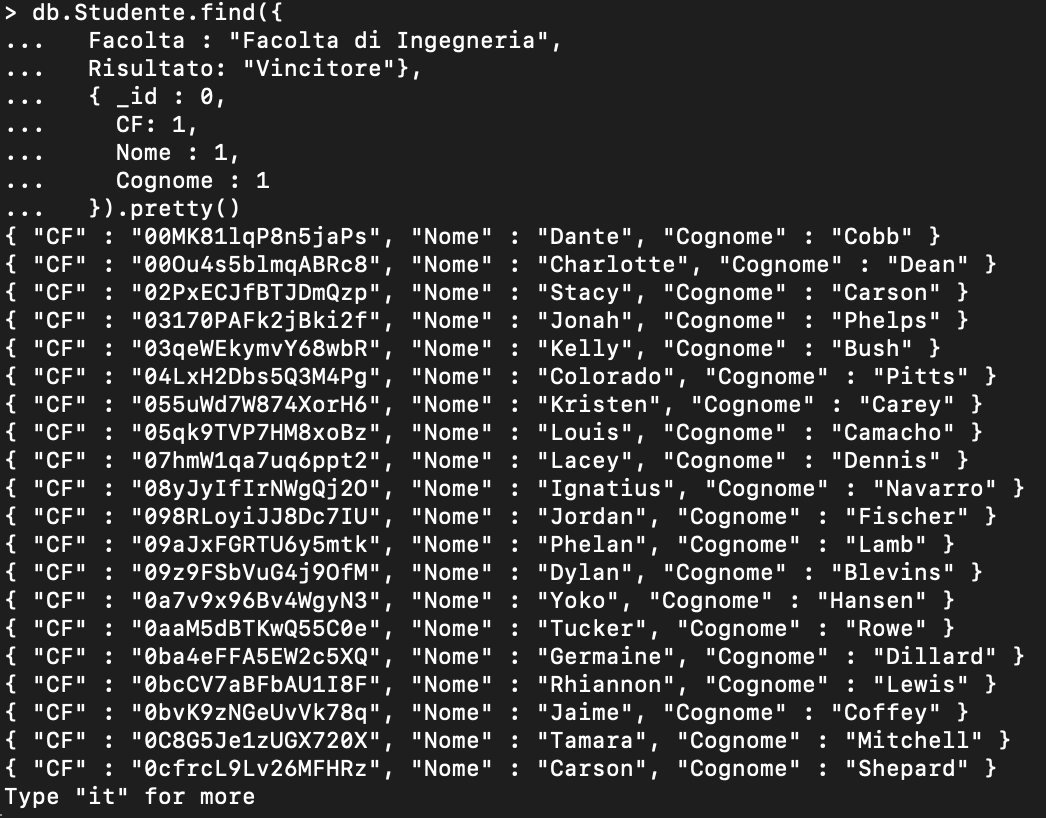
**DELETE**

L’eliminazione in MongoDB avviene attraverso il comando remove.



**QUERY MONGODB**

1. **Selezionare tutti gli studenti iscritti a un corso della ‘Facolta di Ingegneria’ vincitori di una borsa di studio Erasmus**

****

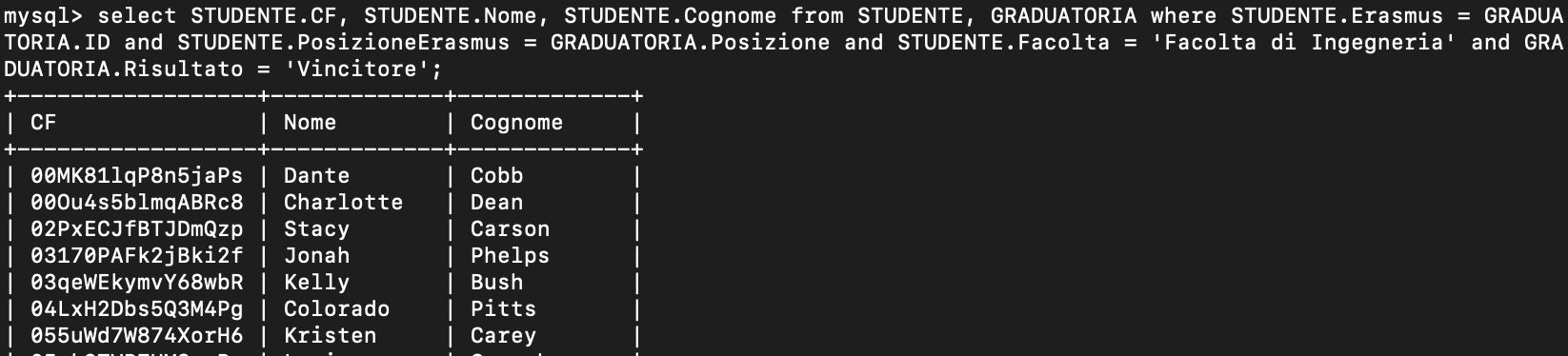
1. **Selezionare tutti gli studenti che sono partiti dopo il 2019/11/01 e sono risultati “Vincitori” ordinandoli per CF.**

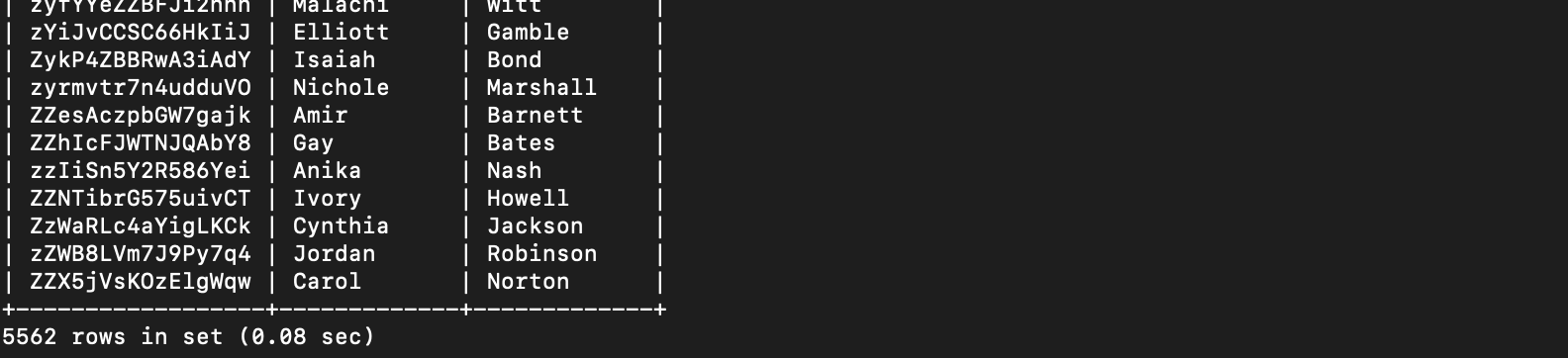
****

**Confronto MongoDB vs MySQL**

1. **La query 1 scritta sopra:**

**MySQL->**

****

****

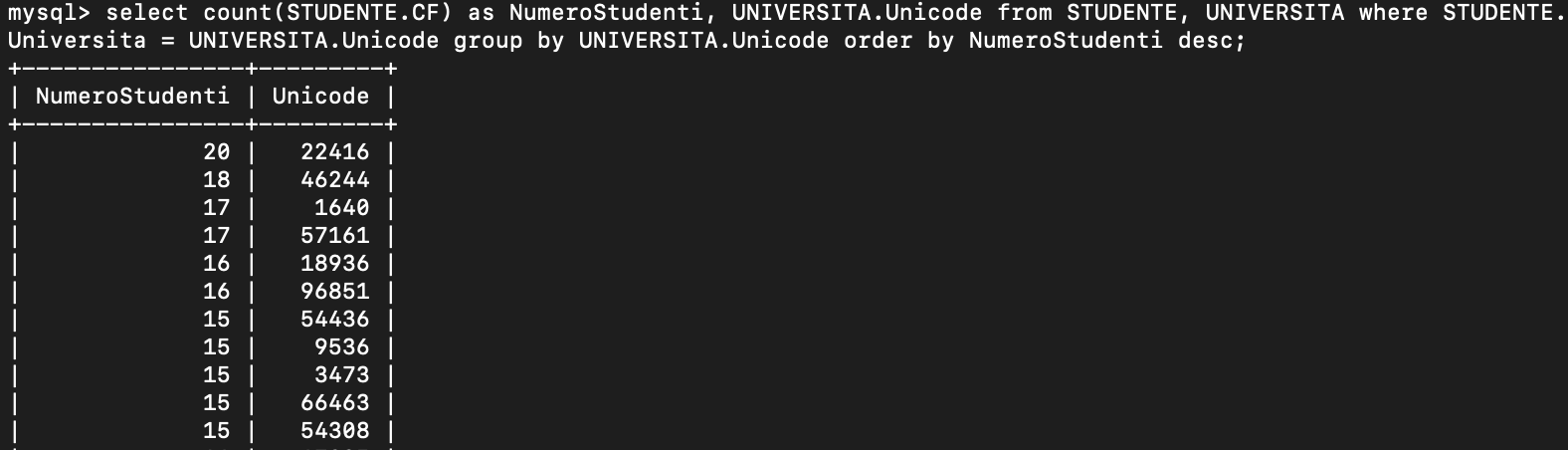
**MongoDB->**

****

MongoDB impiega 0.009 sec contro i 0.08 sec di MySQL.

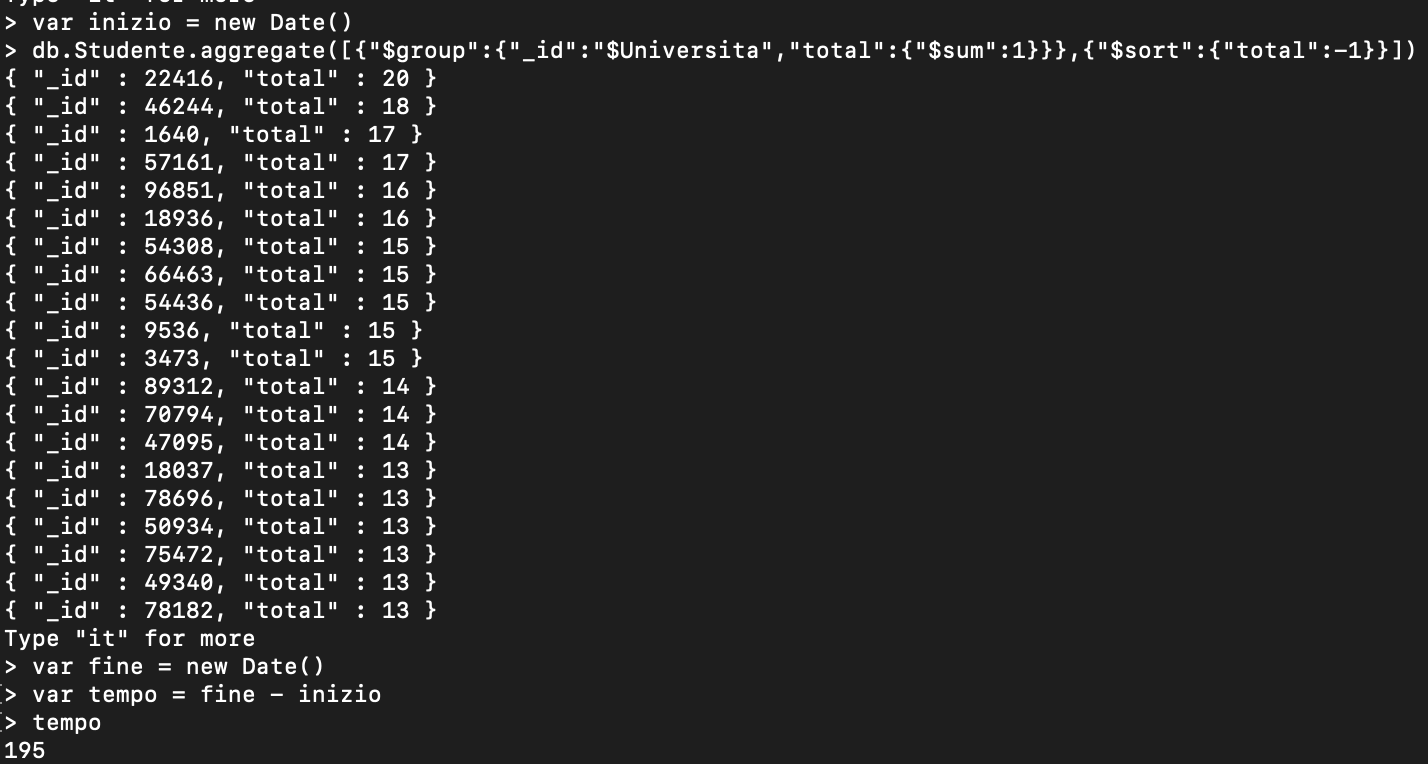
1. **Contare il numero di studenti per ogni Universita ordinando dal più grande al più piccolo**

**MySQL->**

****

****

**MongoDB->**

****

MongoDB impiega 0.195 sec contro i 0.14 sec di MySQL.