UNIVERSITÀ DEGLI STUDI DI NAPOLI PARTHENOPE

SCUOLA INTERDIPARTIMENTALE DELLE SCIENZE, DELL'INGEGNERIA E DELLA SALUTE

INFORMATICA
CORSO DI BASI DI DATI E LABORATORIO

esse4

Proponenti:

Mungari Alfredo	0124002134
Giordano Orsini Massimiliano	0124002214
Caruso Denny	0124002062

Data di Consegna:
17/07/2021

Anno Accademico:
2020 – 2021

Categoria:
Portale Studenti

Indice

Elenco delle figure

Elenco delle tabelle

Elenco degli script

Bibliografia

[1] Ramez A. Elmasri, Shamkant B. Navathe, “Sistemi di basi di dati – fondamenti e complementi”, Pearson, 2018.

Progettazione

Requisiti
Glossario
E/ER
Relazionale
Utenti
Operazioni
Volumi
Vincoli
Normalizzazione
Estensioni

In questo capitolo si tratta la progettazione del database. I vari requisiti sono raccolti, dissezionati e documentati in linguaggio naturale, con l'ausilio di vari diagrammi e tabelle.

Progettazione

”Calcolate la stima migliore del tempo che serve per realizzare un progetto, se tutto va bene dovrete moltiplicarla per ventidue, altrimenti per ventiquattro.“

- riformulazione dei proponenti del progetto esse4

Si vuole realizzare la gestione di un portale studenti. A partire da profonde analisi, discussioni e conversazioni, sono scaturiti i requisiti seguenti.

Sintesi dei requisiti

Il database deve riprodurre (per quanto possibile) il funzionamento di un noto portale studenti che permette di fornire i cosiddetti “Servizi Informatici” per Studenti e Docenti. In particolare, occorre tenere traccia dei seguenti aspetti:

- Studenti
  - pagamento delle tasse
  - partecipazione seminari
  - prenotazione ricevimenti
  - esami superati
  - edizioni degli insegnamenti frequentati
  - compilazione questionari edizioni insegnamenti
  - prenotazione appelli
  - prenotazione e conseguimento borse di studio
  - prenotazione e conseguimento progetti Erasmus
  - partecipazione seminari
  - prenotazione appelli seduta di laurea, lauree conseguite
  - conseguimento tirocini formativi

- Docenti
  - organizzazione seminari
  - tutoraggio tirocini
  - relatori studenti
  - pianificazione ricevimenti studenti
  - pianificazione appelli
  - pianificazione appelli seduta di laurea
  - edizioni dell’insegnamento in cui è coinvolto e per i quali vien valutato

- Corsi di Laurea
  - insegnamenti proposti e edizioni degli insegnamenti attualmente attive
  - studenti iscritti e laureati

Gli studenti devono essere in grado di connettersi al database, visualizzare i dettagli delle tasse, effettuare il loro pagamento, visualizzare eventuali more dovute a ritardi nei pagamenti, nonché la conferma di avvenuto pagamento.
Uno studente collegato al portale studenti ha la possibilità di:

- partecipare a un seminario tenuto da un docente (prenderemo in esame successivamente il ruolo di un docente generico). Ciò è possibile solo se non è stata raggiunta la capienza massima di studenti prevista per quel seminario. Ogni seminario ha un nome, una data e un'ora di inizio. Uno studente in seguito alla partecipazione ad un seminario riceve un certo numero di CFU (Crediti Formativi Universitari). Un seminario non può esistere se non esiste il docente organizzatore

- effettuare una prenotazione per un ricevimento con un determinato docente. All'atto della prenotazione, viene stabilito il turno dello studente sotto forma di numero e la data in cui è avvenuta la prenotazione. Ogni ricevimento di un docente ha una data e un'ora di inizio prefissata, nonché una durata prevista approssimativa. Un ricevimento dipende strettamente dal docente che riceve

- compilare il questionario relativamente a un docente per una determinata edizione dell'insegnamento che l'ha visto come insegnante di un certo tipo (per esempio di teoria e/o di laboratorio). Il questionario riguarda diversi aspetti di valutazione come il rispetto dell'orario delle lezioni da parte del docente, la sua disponibilità, il gradimento generale dell'edizione dell'insegnamento insegnata dal docente e la qualità del materiale didattico fornito dal docente per quella specifica edizione di un insegnamento. Prima di poter sostenere un esame, uno studente deve compilare il questionario. Inoltre, studenti fuoricorso non possono compilare nuovamente il questionario se frequentano di nuovo il corso

- prenotarsi per più appelli di una determinata edizione di insegnamento. Ogni appello è contraddistinto dal codice dell'insegnamento, dall'anno accademico e dalla data dell'appello. Inoltre, presenta alcune informazioni come la data di inizio iscrizione, la data di fine iscrizione, il numero degli studenti consentiti per quell'appello e il tipo di appello (prova scritta, orale, etc.). Uno studente che effettua una prenotazione per un appello riceve un numero di prenotazione e viene memorizzata la data di avvenuta prenotazione. Non si gestiscono appelli riservati per particolari categorie di studenti (es. appelli riservati solo ai fuoricorso)

- frequentare più edizioni insegnamento. Un'edizione insegnamento è identificata univocamente grazie ad un codice di insegnamento e ad un anno accademico di riferimento. Inoltre, presenta diverse informazioni di cui si vuole tenere traccia come i CFU, le modalità di svolgimento di quella determinata edizione dell'insegnamento, l'orario delle lezioni, il semestre e l'anno di corso lungo il quale viene erogato. Un insegnamento generico invece è semplicemente identificato dal suo codice e inoltre fornisce informazione sul nome di quell'insegnamento. La differenza tra insegnamento e edizione insegnamento sta nel fatto che l'edizione insegnamento fa riferimento a quello specifico anno accademico e non all'insegnamento in quanto corso caratterizzante di uno specifico corso di laurea. Si è fatto questo tipo di scelta per modellare la possibilità del docente di insegnare delle edizioni di un insegnamento e non l'insegnamento in sé. Inoltre, i questionari e gli appelli, così come i docenti hanno tra loro una correlazione semantica molto più stretta con edizione insegnamento, piuttosto che con l'insegnamento generico

- può superare o meno un esame, in seguito alla partecipazione ad un appello. In tal caso si vuole memorizzare il numero di verbale, il voto e l'eventuale lode associata all'esame superato dallo studente. Uno studente è iscritto a un solo corso di laurea per volta con una determinata matricola identificante. Infatti, si presuppone che anche al variare della matricola per un passaggio di corso triennale a uno magistrale, la matricola cambi e lo studente venga memorizzato come un nuovo studente, con una nuova matricola che di conseguenza è associata ad un solo corso di laurea, con l'eventuale convalida più o meno corposa degli esami svolti nella sua precedente carriera universitaria. Invece, è chiaro che è ad un corso di laurea possono essere iscritti più studenti. Di un corso di laurea ci interessa memorizzare il nome, la capienza in termini di numero di studenti ammessi, il tipo di laurea cioè se triennale o magistrale e il codice identificativo del corso di laurea. Un corso di laurea offre un certo numero di insegnamenti che possono essere insegnamenti principali come un insegnamento di "Basi di Dati e Laboratorio di Basi di Dati" per il Corso di Studi di "Informatica" oppure "affini/integrativi" come un insegnamento di "Economia e Organizzazione Aziendale" per lo stesso Corso di Studi

- effettuare più tirocini come attività obbligatoria prevista per il suo piano carriera al fine di raggiungere una determinata quota di CFU previsti per il suo Corso di Studi. Un tirocinio è identificato dal numero di tirocinio ed è necessario tenere traccia del numero di CFU erogati con quel tirocinio, della data di inizio e fine del tirocinio stesso. Un tirocinio può essere svolto con l'Università o con un'azienda presente nell'insieme di aziende previste dall'Università presso le quali è possibile svolgere il tirocinio. Un tirocinio ha un tutor aziendale e un tutor in quanto docente. Nel primo caso si tratta di una figura presente nel contesto aziendale di cui è necessario conoscere il numero tesserino, l'anagrafica e lo stipendio. Inoltre, un tutor aziendale fa capo a un'azienda presso la quale è stato assunto in una determinata data e per un certo periodo. Dell'azienda alla quale fa capo il tutor aziendale, è necessario memorizzare la Partita IVA, il nome e l'indirizzo.

- fare domanda per più bandi Erasmus, ma non è detto che ne vinca uno. Quando partecipa a uno di essi, viene memorizzata la data di domanda, mentre quando ne viene assegnato uno a tale studente, si vuole conoscere la data di assegnazione, la data di partenza, di rientro, l'università di destinazione e la località. Infine, di un bando Erasmus, si vuole conoscere il codice identificativo del bando, la data di emissione, la scadenza, il numero di CFU che vengono conseguiti vincendo quel bando e il numero di posti disponibili. Analogamente, uno studente può fare richiesta di più borse di studio, ma non è detto che gli venga assegnata. Quando questi partecipa a un bando di borsa di studio, viene memorizzata la data di domanda, mentre quando ne viene assegnata una a tale studente, si memorizza la data di assegnazione. Infine, di un bando di borsa di studio, si vuole tenere traccia del codice identificativo del bando, il tipo di borsa di studio per il quale si concorre, la causale, il valore netto in termini di denaro, la data di emissione del bando, la scadenza e il numero di borse di studio disponibili.

- può prenotarsi ad un appello di seduta di laurea, al termine della sua carriera universitaria. All'atto della prenotazione, lo studente riceve un numero di prenotazione e si memorizza la data di avvenuta prenotazione. Uno studente può prenotarsi a più appelli di seduta di laurea, senza però presentarsi alla seduta di laurea stessa. Per ogni appello di seduta di laurea è essenziale conoscere la data e il codice del corso di laurea di riferimento che permettono di identificare univocamente un appello di seduta di laurea, la data di inizio e fine iscrizione, il tipo e il numero di studenti consentiti. Mentre per una seduta di laurea, sostenuta da uno studente, è necessario conoscere il verbale della seduta di laurea, il voto assegnato, l'eventuale lode e il tipo. In un appello di seduta di laurea possono esservi più sedute di laurea, ognuna relativa ad uno studente.

Al fine della seduta di laurea, uno studente può avere più relatori ed è necessario conoscere la data di inizio e di fine che identifica il periodo lungo il quale un relatore svolge tale ruolo, il tipo e il titolo della tesi. Un relatore non è altro che un docente.

Di un docente si vuole tenere traccia del suo numero di tesserino che permette di identificarlo univocamente e dell'anagrafica.
Un docente ha la possibilità di:

- insegnare più insegnamenti e più edizioni di tali insegnamenti, che fanno riferimento ad anno accademico differenti
- organizzare seminari, senza eventuali vincoli sul numero o sull'argomento trattato
- pianificare ricevimenti oppure appelli. Per quest'ultimi viene adottata una particolare politica che consente di evitare accavallamenti tra appelli relativi ad edizioni di insegnamento e quindi ad insegnamenti relativi a Corsi di Studi per i quali tale docente insegna. I docenti sostengono più ricevimenti in un giorno, magari di diversi insegnamenti e di diversi corsi di laurea.
- essere presidente presso gli appelli che presiede oppure presso gli appelli di seduta di laurea di cui fa parte
- essere assegnato ad un tirocinio interno. In particolare, la politica adottata consiste nell'assegnare il docente che è stato assegnato a meno tirocini
- visionare gli studenti che in un dato momento, sono prenotati per quel ricevimento in un determinato giorno.
- ricoprire il ruolo di relatore per gli studenti interessati, al termine della loro carriera universitaria.

Inoltre, un docente è valutato a partire dai questionari relativi ad una determinata edizione di insegnamento che insegna svolgendo un certo ruolo.

Nel caso una prenotazione di un appello, appello di laurea, ricevimento sia cancellata, lo *slot* diventa nuovamente disponibile. I dati delle prenotazioni sono conservati per un determinato periodo di tempo, dopodiché vengono cancellati.

Le politiche accennate sono implementate mediante procedure, le quali vengono discusse nel dettaglio nella sezione *"Procedure"* del capitolo *"Progettazione"*.

Fatta questa breve sintesi dei requisiti, è necessario valutare gli utenti del DB:

- Amministratore
- Segreteria
- Docenti
- Studenti

Chiaramente ogni tipologia di utente avrà particolari permessi. Questi ultimi sono definiti in maniera precisa e puntuale all'interno della sezione *"Utenti e loro categorie"* del capitolo *"Progettazione"*. Gli studenti devono potersi connettere al database, prenotare ed eventualmente disdire le prenotazioni per ricevimenti, appelli, appelli di laurea e così via, in totale autonomia, senza intervento del docente stesso. Il docente deve poter fissare data, ora e durata dei vari ricevimenti, degli appelli, dei seminari, variare tali informazioni. La segreteria così come l'amministratore hanno privilegi di più ampio respiro come l'inserimento di nuovi bandi di borsa di studio o Erasmus, il caricamento degli esami superati nella carriera di un particolare studente, il caricamento e l'aggiornamento delle informazioni relativi ai corsi di studio, edizioni di insegnamento, e così via.

Analogamente per le procedure e le viste progettate e implementate, vi sono utenti che possono eseguirle o accedervi rispettivamente e altri che invece non possono farlo. Un esempio può essere la procedura di programmazione automatizzata degli appelli dell'anno accademico corrente. Tale procedura può essere eseguita dal docente, dalla segreteria e dall'amministratore del DB, ma non dagli studenti. Nelle sezioni successive vedremo in dettaglio quali permessi hanno a disposizione gli utenti e su quali procedure.

In un contesto reale sono presenti molti vincoli. Alcuni esempi di vincoli implementati sono i seguenti:

- Una prenotazione di un ricevimento, di un appello o altro, non può avvenire in una data antecedente alla data prevista di inizio prenotazioni e dopo la data prevista di fine iscrizioni
- Le e-mail devono rispettare un determinato formato
- Non è possibile pagare una tassa due volte
- Uno studente non può vincere un bando di borsa di studio se ne ha già vinto uno nello stesso anno accademico o se non ha presentato la domanda di partecipazione. Analogamente per i bandi Erasmus

Per quanto riguarda la dimensione stimata delle tabelle a regime risulta essere molto approssimativa in quanto il contesto sebbene ben focalizzato, risulta prendere in considerazione una moltitudine di aspetti. In generale al crescere delle dimensioni dell'università in termini di studenti iscritti all'anno, si può passare da un numero elevato di tuple ad uno estremamente elevato in quasi tutte le relazioni che lo vedono coinvolto. Una stima molto approssimativa per un ateneo di medie dimensioni è riportata nella sezione *"Volumi"* del capitolo *"Progettazione"*. Le prestazioni attese, considerando che non verranno trattati temi come l'ottimizzazione, le strutture di memorizzazione e affini, si possono definire tutto sommato "buone".

Glossario

Il glossario ha lo scopo fondamentale di chiarire il gergo tecnico usato nella descrizione dei requisiti e di evidenziare eventuali sinonimie e omonimie.

Termine	Definizione	Sinonimi	Omonimi
Studente	Persona che può: • iscriversi ad appelli di esami • iscriversi ad appelli di seduta di laurea • iscriversi a ricevimenti, • iscriversi a seminari • frequentare edizioni di insegnamento • pagare tasse • partecipare a bandi di borsa di studio • partecipare a bandi Erasmus • effettuare un tirocinio • compilare un questionario dell'edizione insegnamento frequentata	Ricevendo, prenotato, candidato	-
Docente	Persona che può: • essere valutata dai questionari per una determinata edizione insegnamento da lui tenuta • insegnare delle edizioni di insegnamento • svolgere il ruolo di relatore per gli studenti interessati • organizzare seminari	Ricevente, organizzatore, presidente, professore	-

	• svolgere il ruolo di tutor in un tirocinio di uno studente • partecipare a una seduta di laurea (con ruolo di presidente o meno) • partecipare agli appelli di edizioni insegnamento (con ruolo di presidente o meno)		
Slot	Singolo incontro docente-studente, singola prenotazione appello-studente, singola prenotazione bando-studente. In generale un'unità elementare, parte di un concetto più complesso che prevede più unità elementari.	Incontro, turno	
Matricola	Codice identificativo univoco di uno studente	-	-
Numero Tesserino	Codice identificativo univoco di uno docente e di un tutor aziendale. In generale la loro "costruzione" si suppone differente, in quanto il primo è determinato ed elaborato dall'Università alla quale il docente afferisce. Il secondo invece, è determinato ed elaborato dall'azienda che attualmente ha assunto tale persona.	-	-
Seminario	Particolare corso di studio, anche non universitario, in cui gli studenti partecipano attivamente, intervenendo con relazioni monografiche o prendendo parte a dibattiti. Un seminario universitario può prevedere un numero di CFU agli studenti che vi partecipano. Ha un numero massimo di persone che possono iscriversi, una data e un'ora di inizio.	-	-
CFU	Crediti Formativi Universitari. Rappresentano uno strumento per misurare la quantità di lavoro di apprendimento, compreso lo studio individuale, richiesto allo studente per acquisire conoscenze e abilità nelle attività formative previste dai corsi di studio.	Crediti	-
Relatore	Docente incaricato di presentare o svolgere una relazione. Nella prassi universitaria, ciascuno dei professori che riferiscono ai colleghi intorno alla tesi di laurea presentata da un candidato. Un relatore viene assegnato a uno studente per un certo periodo di tempo, durante il quale lo studente darà frutto a una tesi che avrà un determinato titolo e sarà di un determinato tipo: compilativa o sperimentale. Per poter avere un relatore, lo studente deve aver superato gli esami fondamentali per preparare la tesi di laurea.	-	-
Presidente	Docente incaricato di svolgere il ruolo di presidente in un appello di seduta di laurea o in un appello di un'edizione insegnamento. Svolge ruoli amministrativi, decisionali ed esecutivi nella seduta di laurea o appello cui è presidente.	-	-
Ricevimento	Periodo di tempo lungo il quale un docente "riceve" un determinato studente in seguito a una sua prenotazione.	-	-

	Lo studente prenota un ricevimento con un docente generalmente per avere dei chiarimenti su aspetti poco chiari dell'edizione di insegnamento impartita da quel determinato docente.		
Questionario	Strumento mediante il quale un docente può essere valutato sotto diversi punti di vista nell'ambito dell'edizione insegnamento da lui/lei impartito. Il questionario è compilato dagli studenti che hanno frequentato il corso di quella edizione di insegnamento. Per compilare il questionario lo studente deve aver frequentato l'edizione dell'insegnamento relativo	Questionario di valutazione	-
Laurea	La laurea è un titolo di studio universitario rilasciato tipicamente da un istituto di istruzione superiore, generalmente un'università.	-	-
Corso di Laurea	Il Corso di Laurea rappresenta il I livello degli studi universitari. Per conseguire la Laurea lo studente deve acquisire 180 crediti formativi universitari (CFU) distribuiti in un massimo di 20 prove. Il Corso di Laurea dura tre anni e alla fine del percorso di studi si consegue la Laurea e si ottiene la qualifica di Dottore. L'obiettivo del Corso di Laurea è quello di garantire una preparazione generale relativa alle discipline di base e caratterizzanti il Corso scelto, anche finalizzata alla formazione professionale per favorire l'inserimento nel mondo del lavoro.Ogni Corso di Laurea ha l'obiettivo di formare figure professionale precise. Dopo aver conseguito la Laurea (L) gli studenti hanno il titolo per accedere a: • Master universitari di I livello • Corsi di Alta Formazione, di Aggiornamento permanente o ricorrente • Laurea Magistrale I Corsi di Laurea sono inquadrati in classi ministeriali.	-	CdL
Appello	Nel contesto universitario consiste della chiamata di più persone per nome e cognome, secondo un ordine prestabilito, ai fini di controllarne le conoscenze acquisite e riconoscerne particolari capacità che avrebbe dovuto acquisire durante l'edizione insegnamento dell'appello al quale ha effettuato la prenotazione.	Seduta d'esame, esame	-
Insegnamento	Un insegnamento è un corso presente nel Corso di Laurea scelto dallo studente. Un insegnamento ha un nome e un codice. Un insegnamento può avere più docenti che lo insegnano nel corso degli anni accademici, può variare nel numero di CFU, nelle modalità di svolgimento, può essere spostato in un anno del corso di studi differente	Corso	-

	per un determinato corso di laurea o in un semestre differente, può avere orari di lezioni differenti nel corso degli anni e a seconda del corso di studi eventualmente.		
Edizione Insegnamento	Un'edizione insegnamento è l'insegnamento durante un determinato anno accademico. Ragion per cui un'edizione insegnamento ha un orario lezioni prefissato, un numero di CFU che permette di conseguire prefissato, una modalità di svolgimento prefissata e così via per gli altri aspetti non prefissati di un insegnamento. Un'edizione insegnamento può prevedere che vi siano più docenti ad insegnarla, ma ognuno con un ruolo ben preciso (teoria, laboratorio, entrambi).	Corso	-
Esame Superato	Un esame è un appello. Un esame superato è un appello andato a buon fine, cioè lo studente prenotato per quell'appello, ha superato l'esame. Lo studente può superare l'esame solo se raggiunge un voto maggiore o uguale a 18 (diciotto) e gli si può essere assegnata la lode se e solo se ci sono le circostanze necessarie a farlo e se ha raggiunto un voto pari a 30 (trenta). Si tiene traccia di ogni esame superato da uno studente mediante un numero di verbale univoco.	-	-
Bando Erasmus e di borsa di studio	Un Bando nel contesto universitario prevede un codice identificativo numerico, un regolamento, dei requisiti, una tipologia, una data di emissione e così via. Nel contesto universitario analizzato ci sono solo 2 (due) tipi di bandi: • Erasmus, acronimo di EuRopean Community Action Scheme for the Mobility of University Students, è un programma di mobilità studentesca dell'Unione europea, nato il 15 giugno 1987. • Borsa di studio, un finanziamento agli studi che viene concesso a studenti che non dispongano di adeguato sostegno economico o che si sono dimostrati particolarmente meritevoli durante il loro percorso di studi universitario. Uno studente può candidarsi per partecipare a un bando Erasmus e/o di borsa di studio. Solo gli studenti che hanno presentato la candidatura per parteciparvi possono vincerne uno. Inoltre, si prevede che uno studente non possa vincere più bandi di borsa di studio per anno accademico. In maniera analoga si prevede che uno studente non possa vincere più bandi Erasmus per anno accademico.	Concorso	-
Tassa	Tributo pagato all'università o alla regione dagli studenti per iscriversi all'università, per usufruire dei servizi universitari, per migliorarli, etc.	Rata	-

	La tassa ha una scadenza, è di un determinato importo e al momento del pagamento viene generato un codice IUV. Se la tassa viene pagata una volta superata la scadenza, allora va pagata anche una mora.		
IUV	L'Identificativo Univoco di Versamento (IUV) è un numero, conforme per formato agli standard stabiliti da AgID (Agenzia per l'Italia Digitale), può essere generato solo dall'Ente creditore. Il codice IUV è univoco: ad un pagamento può corrispondere uno ed uno solo IUV.	-	-
Seduta di laurea	La prova finale, denominata "seduta di laurea" consiste nella stesura e nella discussione pubblica di un elaborato scritto in lingua italiana o straniera, che consentirà allo studente, sotto la guida di un relatore, di dimostrare le competenze teoriche e tecniche raggiunte, la sua capacità di elaborazione critica e di reperimento e uso appropriato delle fonti bibliografiche e documentali.	Esame di laurea, prova finale	-
Appello di seduta di laurea	Concetto astratto che rappresenta la seduta di laurea prima che abbia luogo. Un appello di seduta di laurea ha un massimo numero di studenti consentiti e un periodo lungo il quale gli studenti con i giusti requisiti possono iscriversi (numero di CFU raggiunti, relatore, etc.)		
Tirocinio	Il tirocinio curriculare è un'esperienza formativa che uno studente svolge presso una struttura convenzionata con l'Università (ente o azienda) per conoscere direttamente il mondo del lavoro. Il tirocinio può essere obbligatorio o facoltativo, secondo quanto determinato dal singolo corso di studio, e deve essere svolto nell'anno di corso previsto nel piano didattico. Al tirocinio previsto nei piano di studi è riconosciuto un numero di crediti formativi universitari (CFU). Un tirocinio prevede un tutor aziendale o un tutor docente che seguiranno lo studente impegnato nel tirocinio in tutte le sue fasi.	-	-

Diagramma E/ER

Il diagramma E/ER è visualizzabile sia all'interno del file allegato "diagramma_eer_esse4", che qui di seguito (pagina 11). Per una migliore di visualizzazione, si consiglia di visualizzare il diagramma aprendo il file esterno. Si noti che non sono indicate le cardinalità massime e minime delle associazioni e che la scelta di non realizzare la specializzazione di *Persona* è dovuta alla volontà di aderire maggiormente a quelli che sono i DB effettivamente implementati come portali studenti. Infatti, usando questo tipo di approccio, non è necessario tenere traccia dei cambiamenti di carriera degli studenti nel corso del tempo, per i quali quando inizia una nuova carriera universitaria o eventualmente un'iscrizione a nuovi corsi di laurea, allora la matricola dello studente e ne riceve una nuova. Si noti inoltre, l'assenza del campo *codice fiscale* sulle *entità studente, docente, tutor aziendale*. La sua assenza viene motivata nella discussione delle forme normali.
Si precisa che la notazione usata è quella del libro "Sistemi di basi di dati – fondamenti e complementi", Ramez A. Elmasri, Shamkant B. Navathe.

Diagramma relazionale

Il diagramma relazionale è visualizzabile sia all'interno del file allegato "diagramma_relazionale_esse4", che qui di seguito (pagina 12). Per una migliore di visualizzazione, si consiglia di visualizzare il diagramma aprendo il file esterno.
Si fa notare che tutte le coppie di attributi data e ora riferite allo stesso evento sono state tradotte in un campo unico, in vista dell'implementazione nel DBMS scelto. Nel nostro caso il DBMS scelto è quello di Oracle (in Oracle il tipo date include anche l'ora). Inoltre, si noti come alcuni degli attributi presenti nel diagramma E/ER, non vengano riportati nel diagramma relazionale perché verranno determinati mediante trigger e viste.

Diagramma E/ER

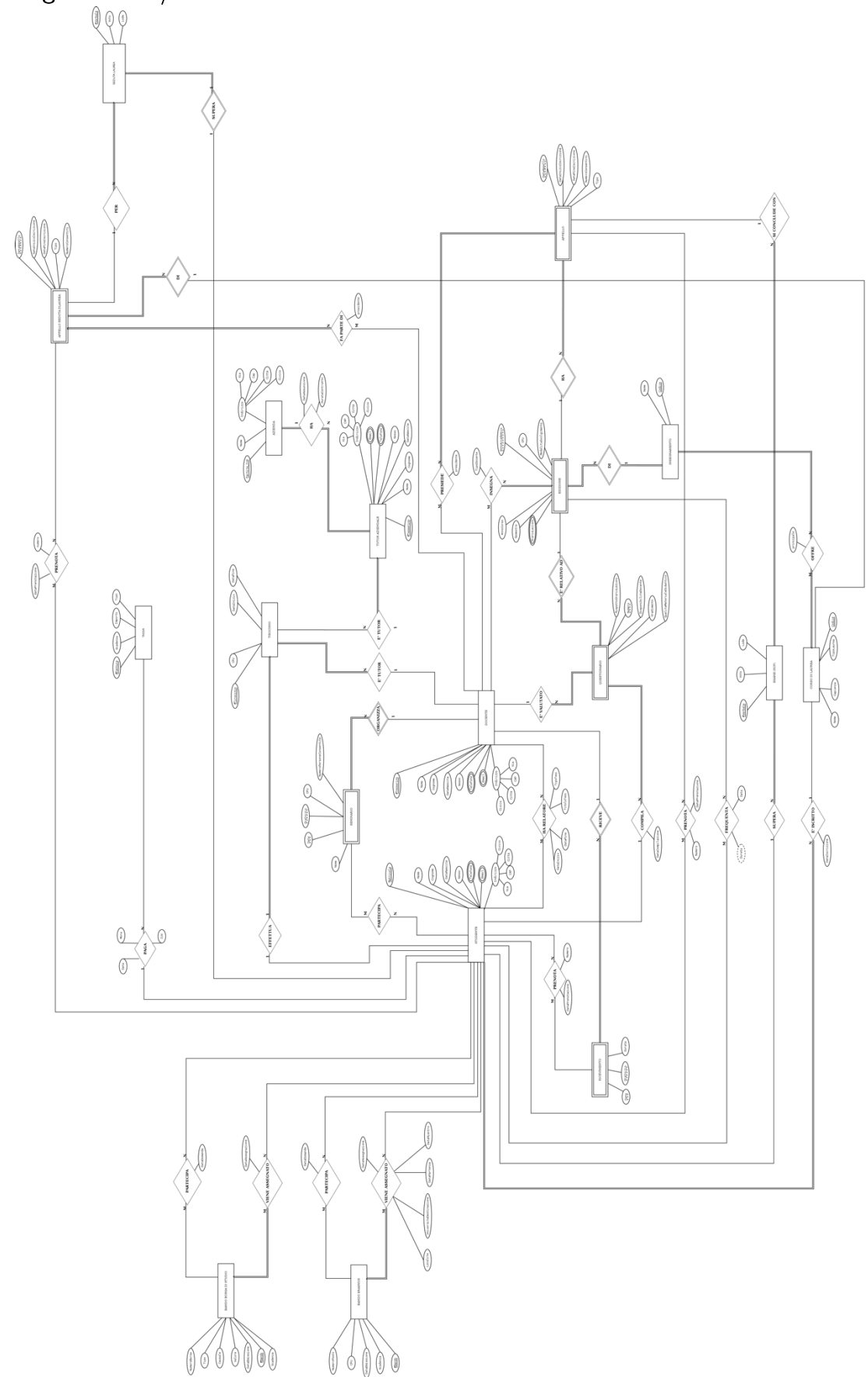
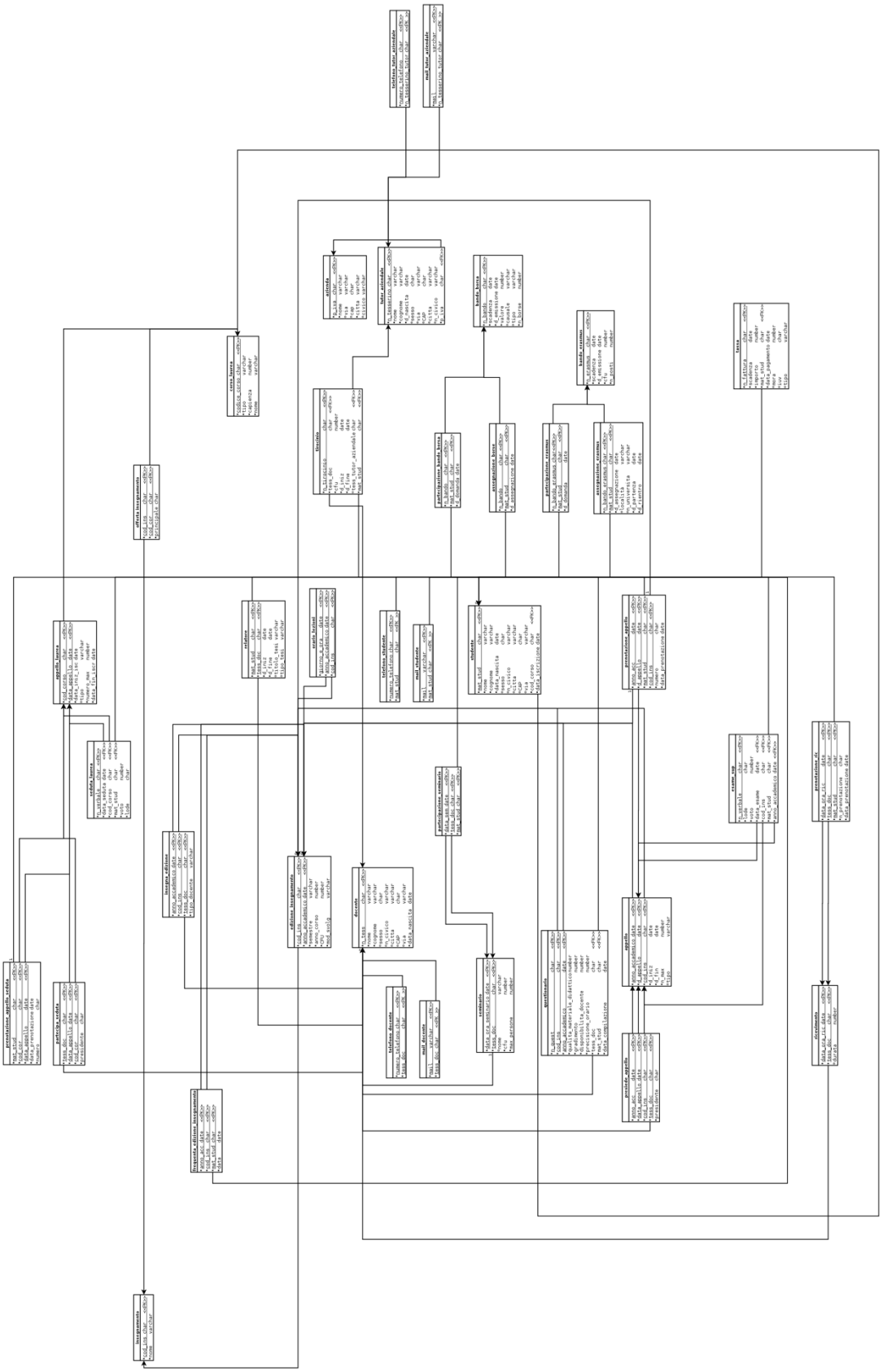


Diagramma Relazionale



Utenti e loro categorie

Nel contesto preso in considerazione dal nostro DB potrebbero essere presi in considerazione una moltitudine di utenti, così come avviene nei casi reali. Si sceglie di semplificare il modello ostico e complesso, esaminando solo gli utenti principali che in tale modello reale prendono parte attiva a un DB del genere. In particolare, ci si riferisce agli studenti, i docenti, la segreteria e infine, un utente amministratore (admin). Dato lo scenario semplificato, non occorrono ruoli per gestire i permessi ed è sufficiente un uso moderato delle viste.
La tavola degli utenti è riportata qui di seguito. Si fa notare che non sono indicati i privilegi di sistema, come il CONNECT, ma solo quelli di oggetto.

Utente	Tipo	Volume	Permessi
DB_esse4	Amministratore	1	ALL PRIVILEGES
Docente	Comune	1	SELECT ON corso laurea SELECT ON studente SELECT ON insegnamento SELECT ON offerta insegnamento SELECT ON edizione insegnamento SELECT, INSERT, UPDATE, DELETE ON frequenta edizione insegnamento SELECT ON appello laurea SELECT ON prenotazione appello seduta SELECT ON partecipa seduta SELECT ON seduta laurea SELECT ON esame superato SELECT ON relatore SELECT ON questionario SELECT, INSERT, UPDATE ON appello SELECT ON orario lezioni SELECT, INSERT, UPDATE ON seminario SELECT ON presiede appello SELECT, INSERT, UPDATE ON ricevimento SELECT ON prenotazione ricevimento SELECT ON prenotazione appello SELECT ON telefono docente SELECT ON email docente SELECT ON email studente SELECT ON email tutor aziendale SELECT ON docente SELECT ON partecipa seminario SELECT ON tirocinio SELECT ON azienda SELECT ON tutor aziendale SELECT ON insegna edizione SELECT ON vista corsi di laurea posti disponibili SELECT ON vista seminari prossimo mese EXECUTE ON procedura programmazione appelli

Studente	Comune	1	SELECT ON corso laurea SELECT ON docente SELECT ON insegnamento SELECT ON edizione insegnamento SELECT ON offerta insegnamento SELECT ON insegna edizione SELECT ON frequenta edizione insegnamento SELECT ON appello laurea SELECT ON partecipa seduta SELECT ON seduta laurea SELECT ON azienda SELECT ON tutor aziendale SELECT ON tirocinio SELECT ON appello SELECT ON orario lezioni SELECT ON seminario SELECT ON presiede appello SELECT ON ricevimento SELECT ON prenotazione ricevimento SELECT ON prenotazione appello SELECT ON partecipa seminario SELECT ON bando borsa SELECT ON bando erasmus SELECT ON assegnazione borse SELECT ON assegnazione erasmus SELECT ON email studente SELECT ON email docente SELECT ON email tutor aziendale SELECT ON vista bando borsa studio SELECT ON vista bando erasmus SELECT ON vista corsi di laurea posti disponibili SELECT ON vista seminari prossimo mese EXECUTE ON procedura calcola statistiche esami e laurea
Segreteria	Comune	1	SELECT, INSERT, UPDATE, DELETE ON corso laurea SELECT, INSERT, UPDATE, DELETE ON studente SELECT, INSERT, UPDATE, DELETE ON docente SELECT, INSERT, UPDATE, DELETE ON insegnamento SELECT, INSERT, UPDATE, DELETE ON edizione insegnamento SELECT, INSERT, UPDATE, DELETE ON offerta insegnamento SELECT, INSERT, UPDATE, DELETE ON insegna edizione SELECT, INSERT, UPDATE, DELETE ON frequenta edizione insegnamento SELECT, INSERT, UPDATE, DELETE ON appello laurea SELECT, INSERT, UPDATE, DELETE ON prenotazione appello seduta SELECT, INSERT, UPDATE, DELETE ON partecipa seduta SELECT, INSERT, UPDATE, DELETE ON seduta laurea SELECT, INSERT, UPDATE, DELETE ON relatore SELECT, INSERT, UPDATE, DELETE ON azienda SELECT, INSERT, UPDATE, DELETE ON tutor aziendale SELECT, INSERT, UPDATE, DELETE ON tirocinio SELECT, INSERT, UPDATE, DELETE ON questionario SELECT, INSERT, UPDATE, DELETE ON appello SELECT, INSERT, UPDATE, DELETE ON orario lezioni SELECT, INSERT, UPDATE, DELETE ON seminario SELECT, INSERT, UPDATE, DELETE ON presiede appello SELECT, INSERT, UPDATE, DELETE ON ricevimento SELECT, INSERT, UPDATE, DELETE ON esame superato

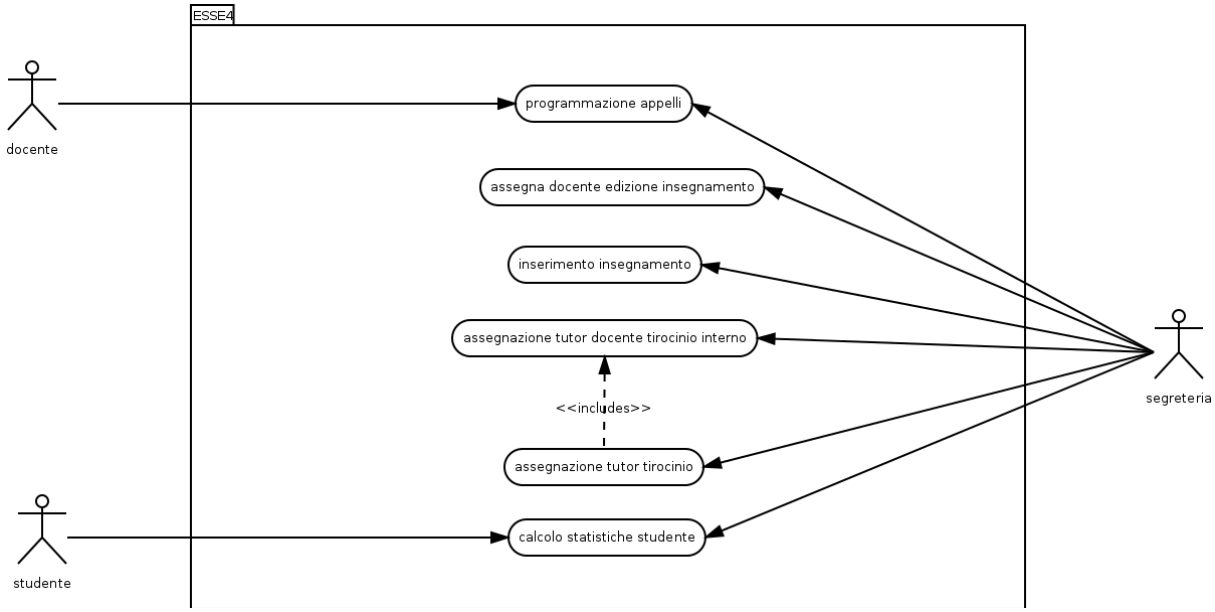
			SELECT, INSERT, UPDATE, DELETE ON prenotazione ricevimento SELECT, INSERT, UPDATE, DELETE ON prenotazione appello SELECT, INSERT, UPDATE, DELETE ON tassa SELECT, INSERT, UPDATE, DELETE ON partecipa seminario SELECT, INSERT, UPDATE, DELETE ON bando borsa SELECT, INSERT, UPDATE, DELETE ON bando erasmus SELECT, INSERT, UPDATE, DELETE ON partecipazione bando borsa SELECT, INSERT, UPDATE, DELETE ON partecipazione bando erasmus SELECT, INSERT, UPDATE, DELETE ON assegnazione borse SELECT, INSERT, UPDATE, DELETE ON assegnazione erasmus SELECT, INSERT, UPDATE, DELETE ON telefono studente SELECT, INSERT, UPDATE, DELETE ON email studente SELECT, INSERT, UPDATE, DELETE ON telefono tutor aziendale SELECT, INSERT, UPDATE, DELETE ON email tutor aziendale SELECT, INSERT, UPDATE, DELETE ON telefono docente SELECT, INSERT, UPDATE, DELETE ON email docente SELECT ON vista corsi di laurea posti disponibili SELECT ON vista seminari prossimo mese SELECT ON vista bando borsa studio SELECT ON vista bando erasmus EXECUTE ON procedura programmazione appelli EXECUTE ON procedura calcola statistiche esami e laurea EXECUTE ON procedura assegnazione tutor tirocinio EXECUTE ON procedura assegnazione tutor docente tirocinio interno EXECUTE ON procedura inserimento insegnamento EXECUTE ON procedura assegnazione docente insegna edizione

Operazioni degli utenti

Oltre alle classiche operazioni di base eseguibili con un comando DML, ogni utente può eseguire determinate operazioni avanzate che sono state implementate mediante procedure. Nel caso del DB *esse4*, le procedure implementate sono le seguenti:

- programmazione automatizzata appelli
- assegnazione docenti in edizione insegnamento
- inserimento di un insegnamento
- assegnazione di un tutor interno per un tirocinio interno
- assegnazione di un tutor interno e un tutor aziendale (esterno) per un tirocinio esterno
- calcolo statistiche studente

Qui di seguito sono riportate le operazioni degli utenti avanzate che sono state implementate e i relativi utenti che le sfruttano mediante il diagramma UML dei casi d'uso. Successivamente sono mostrate le schede descrittore operazioni delle operazioni implementate.



Si precisa che gli utenti sono rappresentati con le immagini stilizzate ai lati mentre le operazioni sono rappresentate al centro. Per quanto riguarda le frecce, indicano la possibilità di un determinato utente di eseguire la procedura a cui punta la freccia. Si adotta la notazione tale per cui, una procedura che invoca un'altra procedura esistente è collegata a quest'ultima mediante una freccia tratteggiata.
Una scheda descrittore operazioni specifica in dettaglio cosa fa una determinata operazione. Ogni operazione avrà una propria scheda descrittore operazione. All'interno di ognuna si troveranno le seguenti informazioni relative all'operazione:

- nome operazione
- scopo
- argomenti, indica i parametri passati all'operazione
- risultato
- errori, indica gli eventuali errori che possono presentarsi durante l'esecuzione dell'operazione
- usa, indica le tabelle usate
- modifica, indica le tabelle modificate
- prima, indica la situazione prima dell'avvenuta esecuzione della procedura
- poi, indica la situazione dopo l'avvenuta esecuzione della procedura

Nome Operazione	programmazione appelli
Scopo	automatizzare la programmazione degli appelli di una determinata edizione di insegnamento dell'anno accademico corrente in merito a un corso di laurea specificato
Argomenti	codice insegnamento, numero studenti massimo, tipo di appello, codice corso di laurea
Risultato	aggiunta appelli/errore
Errori	non esiste l'edizione insegnamento per l'anno accademico corrente dell'insegnamento specificato; non è stato possibile stabilire le date di appello per l'anno accademico
Usa	offerta insegnamento, edizione insegnamento, appello
Modifica	appello
Prima	è possibile programmare un appello in quel giorno
Poi	è stato programmato un appello in quel giorno. Altri appelli dello stesso corso di laurea e appartenenti allo stesso anno di corso non possono essere programmati nello stesso giorno

Nome Operazione	assegnazione docente edizione insegnamento
Scopo	inserire una nuova edizione di insegnamento, stabilire quale docente debba essere assegnato ad essa e
Argomenti	codice insegnamento, tipo docente, anno accademico, semstre, anno corso, CFU, svolgimento
Risultato	aggiunta l'edizione insegnamento e l'assegnazione del docente come insegnante di quella edizione di insegnamento oppure errore.
Errori	non esiste docente che può insegnare questa edizione di insegnamento e che ha insegnato le precedenti edizioni insegnamento, non è stato possibile assegnare un docente a tale edizione di tale insegnamento
Usa	edizione insegnamento, insegnamento, questionario, offerta insegnamento, docente, insegna edizione
Modifica	edizione insegnamento, insegna edizione
Prima	non esiste un'edizione insegnamento come quella che si sta richiedendo di inserire e non è noto quale docente assegnare a tale edizione insegnamento
Poi	è stato assegnato un docente come insegnante dell'edizione insegnamento inserita

Nome Operazione	inserimento insegnamento
Scopo	crea un insegnamento e una sua edizione insegnamento all'interno delle rispettive tabelle. Inoltre permette di associare tale insegnamento a un particolare corso di laurea esistente con la possibilità di specificare se è caratterizzante o meno per tale CdL
Argomenti	codice insegnamento, nome insegnamento, anno accademico, semestre, anno corso, CFU, svolgimento, codice corso, principale
Risultato	inserimento dell'insegnamento, dell'edizione insegnamento e della sua offerta per uno specifico corso di laurea; altrimenti errore
Errori	non è stato possibile inserire l'insegnamento, l'edizione insegnamento specificata o l'associazione con il CdL specificato
Usa	insegnamento, edizione insegnamento, offerta insegnamento
Modifica	insegnamento, edizione insegnamento, offerta insegnamento
Prima	non c'è un'edizione insegnamento, un insegnamento e l'offerta di esso in un CdL che portino le stesse informazioni nelle chiavi primarie delle tuple inserite
Poi	è stata inserita un insegnamento, un'edizione insegnamento e la sua offerta all'interno di uno specifico CdL

Nome Operazione	assegnazione tutor docente tirocinio interno
Scopo	si assegna un tutor docente (tirocinio interno) a uno studente che ne ha fatto richiesta in modo tale che fra tutti i docenti che effettuano i tirocini, gli si assegni quello che ha fatto meno tirocini in assoluto
Argomenti	matricola studente
Risultato	si inserisce un tirocinio tale per cui è stato possibile individuare un docente da assegnare come tutor interno. Altrimenti errore
Errori	Non è stato possibile determinare il docente da assegnare come tutor del tirocinio dello studente la cui matricola è passata come parametro di input. Ciò può essere dovuto al fatto che lo studente non abbia ancora raggiunto una soglia prefissata di CFU per fare richiesta di tirocinio a seconda del tipo di corso di laurea al quale è iscritto
Usa	studente, esame superato, insegnamento, edizione insegnamento, corso laurea, docente, tirocinio
Modifica	tirocinio
Prima	allo studente non è associato un tirocinio interno
Poi	allo studente viene assegnato un tirocinio interno

Nome Operazione	assegnazione tutor tirocinio
Scopo	si assegna un tutor aziendale (tirocinio esterno) a uno studente che ne ha fatto richiesta in modo tale che fra tutti i tutor aziendali di un'azienda che effettuano i tirocini, gli si assegni quello che ha fatto meno tirocini in assoluto. Inoltre, si sfrutta la procedura assegnazione tutor docente tirocinio interno per assegnare anche il tutor interno che seguirà lo studente nel suo percorso di tirocinio
Argomenti	matricola studente
Risultato	si inserisce un tirocinio tale per cui è stato possibile individuare un docente da assegnare come tutor interno e un tutor aziendale come tutor esterno. Altrimenti errore
Errori	Non è stato possibile determinare il docente da assegnare come tutor interno del tirocinio dello studente la cui matricola è passata come parametro di input. Oppure non è stato possibile determinare il tutor aziendale da assegnare come tutor esterno del tirocinio
Usa	studente, esame superato, insegnamento, edizione insegnamento, corso laurea, docente, tirocinio, azienda, tutor aziendale
Modifica	tirocinio
Prima	allo studente non è associato un tirocinio esterno
Poi	allo studente viene assegnato un tirocinio esterno

Nome Operazione	calcolo statistiche studente
Scopo	calcola le statistiche degli esami e della laurea per uno studente la cui matricola viene passata in input
Argomenti	matricola studente
Risultato	vengono stampate e restituite le seguenti informazioni (o occorre un errore): • media aritmetica esami • media ponderata esami • media aritmetica laurea • media ponderata laurea • progressione CFU
Errori	non è possibile ottenere le statistiche degli esami dello studente e relative alla sua laurea
Usa	studente, insegnamento, edizione insegnamento, esame superato, corso laurea
Modifica	-
Prima	lo studente non conosce il suo avanzamento di carriera, il suo stato in termini di medie (aritmetica e ponderata) di esami e di medie (aritmetica e ponderata) di laurea
Poi	lo studente viene a conoscenza del suo avanzamento di carriera, il suo stato in termini di medie (aritmetica e ponderata) di esami e di medie (aritmetica e ponderata) di laurea

A questo punto, si procede con il mostrare la tavola delle operazioni. Tale tavola prevede ben quattro colonne:

- nome operazione
- tipo, indica se si tratta di un'operazione '*batch*' (B) oppure '*interactive*' (I). Nel caso di questo DB, le operazioni sono tutte *batch*
- volume, indica il numero di volte che la procedura viene lanciata. Si precisa che occorre contare il numero di volte in cui l'operazione è lanciata, non necessariamente il numero di volte in cui ha successo
- periodo, indica il periodo lungo il quale è stato stabilito il *volume* di cui prima

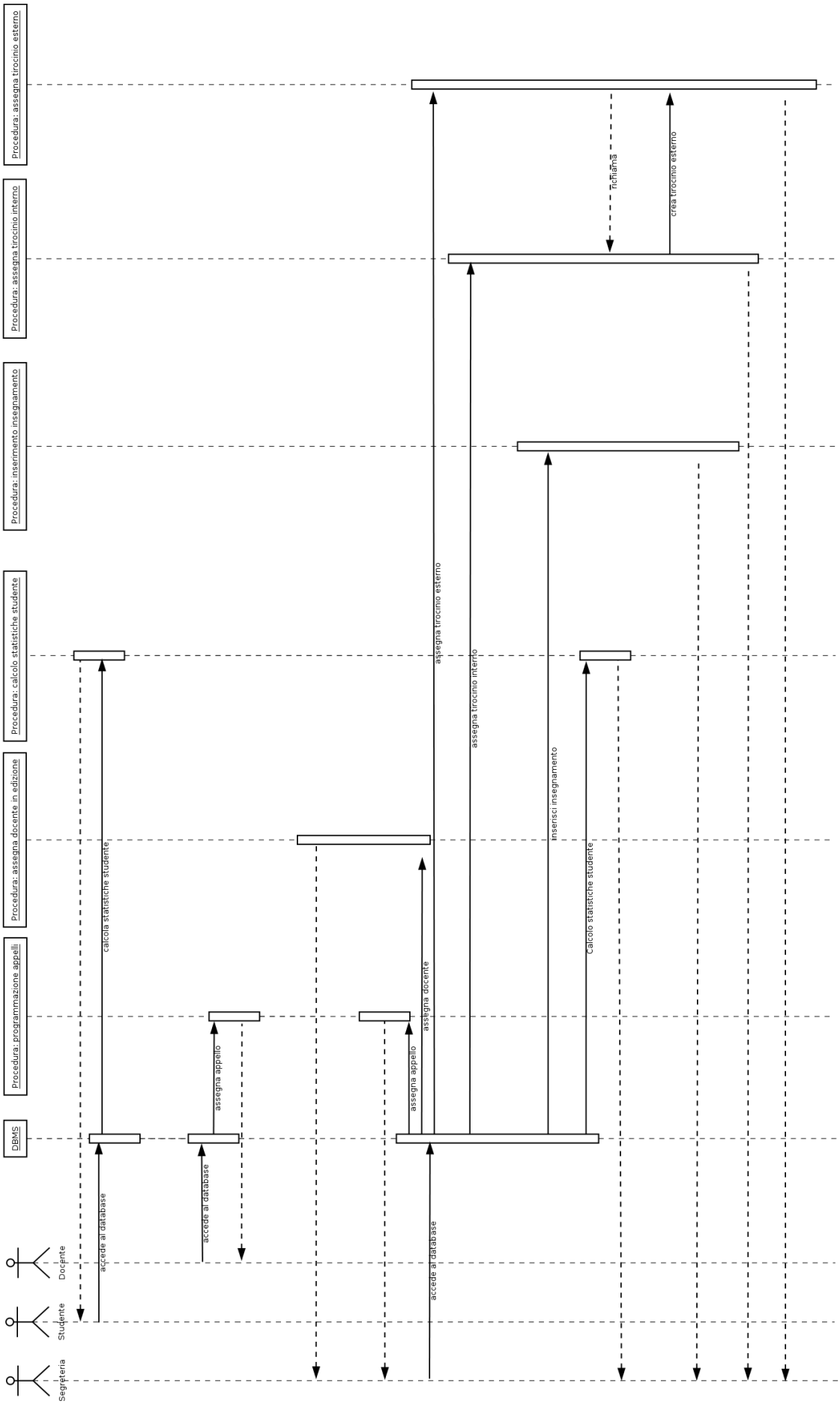
Per stilare la tavola delle operazioni vengono fatte le seguenti considerazioni per cercare di approssimare i volumi effettivi di un sistema reale che sicuramente risulterà essere molto più complesso di quello proposto:

- si suppone che l'ateneo preveda circa 30 corsi di laurea e circa 20 insegnamenti per ciascun corso di laurea e che la procedura di programmazione appelli sia lanciata dal singolo docente una sola volta all'anno
- si suppone che vengano inserite le nuove edizioni insegnamento ad ogni anno accademico e che ogni insegnamento sia affidato a due docenti in media. Inoltre, si suppone che la procedura venga lanciata una volta all'anno per ogni insegnamento
- si suppone che circa 200 studenti per anno effettuino una richiesta di tirocinio che va a buon fine mentre altre 100 non vanno a buon fine (molti tentativi di richiesta tirocinio falliscono). La metà del totale delle richieste di tirocinio sono relative a tirocini interni e l'altra metà a tirocini esterni
- si suppone che uno studente richieda le proprie statistiche circa una volta al mese. Si suppone che l'ateneo accolga circa 15 000 studenti

Fatte tali considerazioni, la tavola delle operazioni è la seguente:

Nome operazione	Tipo	Volume	Periodo
programmazione appelli	B	600	anno
assegnazione docente edizione insegnamento	B	600	anno
inserimento insegnamento	B	1 200	anno
assegnazione tutor docente tirocinio interno	B	150	anno
assegnazione tutor tirocinio	B	150	anno
calcolo statistiche studente	B	15 000	mese

A questo punto è utile capire meglio il funzionamento delle procedure appena elencate, mediante il diagramma UML delle sequenze (detto anche 'UML sequence diagram'). Un diagramma delle sequenze illustra la sequenza dei messaggi tra oggetti in un'interazione. Un diagramma di sequenza consiste in un gruppo di oggetti, rappresentati da lifeline, e nei messaggi che tali istanze si scambiano durante l'interazione. Nel caso del diagramma UML delle sequenze del DB *esse4* (illustrato qui di seguito), si valutano le interazioni tra i vari utenti e le procedure che ognuno di essi può invocare, in maniera tale da illustrare meglio il flusso delle singole operazioni.


Procedura: assegna tirocinio esterno
Procedura: assegna tirocinio interno
Procedura: inserimento insegnamento
Procedura: calcolo statistiche studente
Procedura: assegna docente in edizione
Procedura: programmazione appelli
DBMS
Docente
Studente
Segreteria
richiama
crea tirocinio esterno
assegna tirocinio esterno
assegna tirocinio interno
inserisci insegnamento
Calcolo statistiche studente
calcola statistiche studente
assegna docente
assegna appello
assegna appello
accede al database
accede al database
accede al database

Volumi

La tavola dei volumi è rappresentata qui di seguito. Oltre a riportare il numero verosimile di tuple presenti in ciascuna tabella una volta che il DB sia a regime, rappresenta anche il suo incremento atteso in un periodo di tempo prefissato. Se esiste una politica di aggiornamento dei dati per cui le tuple più vecchie sono rimosse dal DB, oppure se il numero di tuple in una relazione è più o meno costante nel tempo, allora l'incremento è nullo. Si noti che nella seguente figura, la lettera "E" nella colonna "Tipo", sta a significare che quella particolare *tabella* riportata nella stessa riga nella colonna "Tabella", risulta essere un'entità. Per indicare le *tabelle di transizione* (anche dette relazione di relazioni), si usa la lettera "A" che sta per *associazione.* Infine, per indicare le *entità deboli* si usano le due lettere "ED".

Per stilare la tavola dei volumi vengono fatte le seguenti considerazioni per cercare di approssimare i volumi effettivi di un sistema reale che sicuramente risulterà essere molto più complesso di quello proposto:

- Si assume che in media per un ateneo di medie dimensioni (da 10 000 a 20 000 iscritti) vi siano circa 15 000 nuovi iscritti per anno accademico.
- Si assume che in un ateneo di medie dimensioni vi siano circa 30 corsi di laurea e che questo numero resti pressoché costante nel tempo. Inoltre, si assume che per un corso di laurea vi siano circa 30 docenti coinvolti e che questo numero resti pressoché costante nel tempo.
- Si assume che ogni corso di laurea preveda circa 20 insegnamenti e che questo numero resti pressoché costante nel tempo. Mentre per quanto riguarda le edizioni di un insegnamento, queste vengono inserite le nuove e conservate le precedenti. Discorso analogo per le tuple della relazione di relazioni "insegna edizione". Viceversa, per la relazione di relazioni "offerta insegnamento" si fa capo all'insegnamento che resta pressoché costante nel tempo.
- Si assume che vengano mantenute le informazioni relative agli appelli di laurea precedenti, che vengano inseriti circa 30 appelli di laurea al mese (uno per ogni corso di laurea), che le prenotazioni da parte degli studenti possano essere al più 20 per ogni appello di laurea, che vengano cancellate le prenotazioni degli appelli di laurea passati, che per ogni appello di laurea vi siano circa 8 docenti assegnati, che a uno studente vengano assegnati circa 2 relatori.
- Si assume che le aziende presso le quali gli studenti possono effettuare il tirocinio siano al più 100 e che questo numero resti pressoché costante nel tempo, che ogni azienda abbia circa 2 impiegati con la qualifica di tutor aziendale e che questo numero resti pressoché costante nel tempo.
- Si assume che i questionari vengano compilati dagli studenti, ricevuti, elaborati e poi cancellati. Inoltre, si assume che l'università offra un appello al mese per ogni edizione di insegnamento, che i dati degli appelli vengono valutati, elaborati e poi cancellati, che gli orari delle lezioni cambino di anno accademico in anno accademico e che vengano cancellati gli orari delle edizioni insegnamento precedenti.
- Si assume che un docente organizzi in media un seminario per anno accademico, riceva una volta a settimana e che i dati dei ricevimenti vengano valutate, elaborate e poi cancellate, che in media gli studenti che superano un appello di una determinata edizione insegnamento siano circa la metà dei prenotati allo stesso, che un appello sia presieduto in media da 2 docenti, che per ogni appello vi siano prenotati circa 30 studenti. Siccome la mole di dati relativi alle prenotazioni degli appelli è notevole, allora i dati vengono valutati, elaborati e poi cancellati, che ad ogni ricevimento si ricevono circa cinque studenti.
- Si assume che ogni studente paghi circa 4 tasse per anno accademico, che ci siano 2 bandi di borsa di studio e 2 bandi Erasmus per anno accademico, che ad ognuno di essi partecipino circa 50 000 studenti dell'ateneo, ma solo 1000 di essi risulta essere assegnatario, che ogni studente, docente, tutor aziendale abbiano circa 2 telefoni e 2 e-mail.

Tabella	Tipo	Volume	Incremento	Periodo
Corso di Laurea	E	30	0	anno
Studente	E	15 000	15 000	anno
Docente	E	900	0	anno
Insegnamento	E	600	0	anno
Edizione Insegnamento	ED	600	600	anno
Offerta Insegnamento	A	600	0	anno
Insegna Edizione	A	900	900	anno
Frequenta Edizione Insegnamento	A	15 000	15 000	anno
Appello Laurea	ED	30	30	mese
Prenotazione Appello Seduta	A	600	0	mese
Partecipa Seduta	A	250	250	mese
Seduta Laurea	A	600	600	mese
Relatore	A	1200	1200	anno
Azienda	E	100	0	anno
Tutor Aziendale	E	200	0	anno
Tirocinio	E	600	600	anno
Questionario	ED	45 000	0	anno
Appello	ED	600	0	mese
Orario Lezioni	ED	600	0	anno
Seminario	ED	900	900	anno
Presiede Appello	A	1200	0	mese
Ricevimento	ED	3600	0	mese
Esame Superato	E	9 000	9 000	mese
Prenotazione Ricevimento	A	18 000	0	mese
Prenotazione Appello	A	18 000	0	mese
Tassa	E	60 000	60 000	anno
Partecipa Seminario	A	27 000	27 000	anno
Bando Borsa	E	2	2	anno
Bando Erasmus	E	2	2	anno
Partecipazione Bando Borsa	E	100 000	0	anno
Partecipazione Bando Erasmus	A	100 000	0	anno
Assegnazione Erasmus	A	2000	2000	anno
Assegnazione Borse	A	2000	2000	anno
Telefono Studente	ED	30 000	30 000	anno
Telefono Docente	ED	1800	0	anno
Telefono Tutor Aziendale	ED	400	0	anno
Email Studente	ED	30 000	30 000	anno
Email Docente	ED	1800	0	anno
Email Tutor Aziendale	ED	400	0	anno

Vincoli di integrità

Sono detti statici i vincoli che limitano i valori assumibili da alcuni attributi indipendentemente dal tempo, mentre sono detti dinamici quelli che riguardano valori che cambiano nel tempo; anche molte *regole di business* si presentano come vincoli. Poiché tutti i dati sono soggetti ad evoluzione con velocità diverse, non si può essere attaccati alla lettera di queste definizioni, né si può pretendere di produrre un elenco esaustivo dei vincoli pertinenti ad un dato contesto. Si riportano i vincoli più importanti per il DB *esse4*. Non compaiono nell'elenco, in quanto ovvi, i vincoli di chiave primaria e chiave esterna.

Vincoli Statici:

- Un corso di laurea può essere triennale o magistrale. Il numero di studenti che possono iscriversi dev'essere maggiore di zero. Il tipo, la capienza e il nome del corso di laurea devono essere obbligatori.
- Il sesso di uno studente, un docente o un tutor aziendale può essere maschio, femmina o non definito. La data di nascita di uno studente dev'essere precedente alla sua data di iscrizione. Il nome, il cognome, la data di nascita devono essere obbligatori sia per lo studente, sia per il docente che per il tutor aziendale.
- Un'edizione insegnamento può trovarsi al primo semestre, al secondo semestre oppure essere annuale. Può essere afferente al primo, secondo, terzo, quarto o quinto anno di corso. Può essere svolto in presenza o in telematica. Deve provvedere (in caso di superamento dell'attività) un numero di CFU maggiore di 0 e inferiore a 13. Un'edizione di un insegnamento può essere insegnata da più docenti a titolo di docente di teoria, di laboratorio o entrambi. Inoltre, il semestre, l'anno di corso, il numero di CFU, lo svolgimento di un'edizione insegnamento devono essere obbligatori.
- Un insegnamento offerto in un corso di laurea può essere caratterizzante o meno.
- Un appello di laurea deve prevedere un numero di iscritti massimo compreso tra 1 e 20. La data di inizio iscrizione per un appello di seduta di laurea dev'essere precedente alla data di fine iscrizione per lo stesso, la data dell'appello dev'essere successiva alla data di inizio iscrizione e alla data di fine iscrizione. Un appello di laurea può avere luogo in presenza oppure in telematica. Ogni appello di laurea fa riferimento a un corso di laurea in maniera diretta. La data di inizio iscrizione, la data di fine iscrizione e il massimo numero di studenti che possono iscriversi devono essere obbligatori.
- Una prenotazione per un appello di laurea deve avvenire prima che vi sia la data di appello per il quale si effettua la prenotazione. La data dell'appello, quella di avvenuta prenotazione e il numero della prenotazione devono essere obbligatori. Inoltre, il numero della prenotazione dev'essere univoco.
- Un appello di seduta di laurea deve avere obbligatoriamente un presidente. Una seduta di laurea di uno studente invece, deve prevedere un voto di laurea compreso tra 66 e 110 e la lode va assegnata se e solo se il voto assegnato risulta essere 110. Inoltre, il voto dev'essere obbligatorio.
- Un relatore o i relatori assegnati a uno studente elaborano insieme a lui o lei, una tesi che può essere di tipo compilativa o sperimentale. La data di inizio assegnazione del relatore dev'essere precedente alla data di fine assegnazione dello stesso. La data di inizio di assegnazione del relatore e quella di fine devono essere obbligatorie.
- Il nome e la città di un'azienda sono obbligatori.
- Un tirocinio deve prevedere un certo numero di CFU conseguiti compreso fra 1 e 12. La data di inizio del tirocinio dev'essere precedente alla data di fine del tirocinio. Il numero di CFU, la data di inizio del tirocinio, la data di fine del tirocinio devono essere obbligatori.

- Un questionario prevede un insieme di aspetti in base ai quali valutare un docente che ha insegnato una determinata edizione di un determinato insegnamento. Per ogni parametro, il valore della valutazione dev'essere compreso tra 0 e 5. Ogni parametro di valutazione e la data di compilazione devono essere obbligatori.
- Un appello deve prevedere un numero di iscritti massimo maggiore di 0. La data di inizio iscrizione per un appello dev'essere precedente alla data di fine iscrizione per lo stesso, la data dell'appello dev'essere successiva alla data di inizio iscrizione e alla data di fine iscrizione. Un appello può essere di tipo orale, scritto oppure non previsto. Ogni appello fa riferimento a un'edizione di un insegnamento. La data di inizio iscrizione, la data di fine iscrizione e il massimo numero di studenti che possono iscriversi devono essere obbligatori. Un appello deve avere obbligatoriamente un presidente.
- Un seminario deve prevedere un numero di CFU conseguibili maggiore o uguale a zero, un numero di persone massimo consentite maggiore di zero. Il numero di CFU e il numero massimo di persone consentite devono essere obbligatori.
- Un esame superato da uno studente deve prevedere che il voto assegnato allo studente sia compreso tra 18 e 30 e che la lode sia assegnato allo studente se e solo se il voto raggiunto è 30. Il voto e la lode devono essere obbligatori.
- Una prenotazione per un ricevimento deve avere un numero di prenotazione univoco ed obbligatorio, una data di prenotazione obbligatoria e si deve prevedere che la data di prenotazione sia precedente alla data del ricevimento stesso.
- Una prenotazione per un appello deve avvenire prima che vi sia la data di appello per il quale si effettua la prenotazione. La data dell'appello, quella di avvenuta prenotazione e il numero della prenotazione devono essere obbligatori. Inoltre, il numero della prenotazione dev'essere univoco.
- Una tassa deve avere un importo maggiore di zero. La scadenza e l'importo devono essere obbligatori.
- Un bando di borsa di studio deve prevedere un valore di borsa di studio (in denaro) maggiore di zero, un numero di borse previste dal bando maggiore di zero e che la data di emissione del bando sia precedente alla data di scadenza del bando. La scadenza, la data di emissione, il valore e il numero di borse di studio previste nel bando devono essere obbligatori. Un'istanza di partecipazione al bando di borsa di studio prevede che la data di domanda sia obbligatoria. Un'istanza di assegnazione di borsa di studio deve prevedere che la data di assegnazione sia obbligatoria.
- Un bando Erasmus deve prevedere un numero di posti maggiore di zero, un numero di CFU conseguibili maggiore di zero e che la data di emissione del bando sia precedente alla data di scadenza del bando. La scadenza, la data di emissione, il numero di CFU e il numero di posti consentiti devono essere obbligatori. Un'istanza di partecipazione al bando Erasmus prevede che la data di domanda sia obbligatoria. Un'istanza di assegnazione di bando Erasmus deve prevedere che la data di assegnazione sia obbligatoria, che la data di assegnazione sia precedente alla data di partenza e di rientro prevista. La data di partenza prevista inoltre, dev'essere precedente alla data di rientro prevista. La data di assegnazione del bando Erasmus, la data di partenza e quella di rientro devono essere obbligatorie.
- Ogni e-mail di uno studente, di un docente e di un tutor aziendale deve prevedere una particolare costruzione. Nel caso dello studente sono previsti dei caratteri alfabetici rappresentanti il nome, un punto, dei caratteri alfabetici rappresentanti il cognome, una chiocciola e la seguente stringa "studenti.uniparthenope.it". Nel caso del docente e del tutor aziendale, la costruzione delle e-mail è analoga se non per il gatto che la stringa successiva alla chiocciola deve corrispondere alla seguente: "uniparthenope.it".

Vincoli Dinamici:

- Uno studente può prenotarsi solo ad appelli di esami riguardanti il suo corso di laurea. Uno studente può prenotarsi a un appello di un esame se e solo se ha pagato tutte le tasse, non ci sono più prenotazioni per lo stesso appello dello stesso studente, non è stato già superato il numero di iscritti consentiti, se non ha già superato quell'esame, se la prenotazione è avvenuta in data coerente con quelle di inizio e fine prenotazione dell'appello stesso.
- Uno studente può effettuare una prenotazione per un appello di seduta di laurea se e solo se l'appello di seduta di laurea per il quale ha eseguito la prenotazione afferisce al suo CdL, se la somma dei CFU ottenuti con i seminari (al più 3), dei CFU ottenuti con il tirocinio (12), dei CFU ottenuti con i bandi erasmus (al più 12) e quelli ottenuti con gli esami del CdL è maggiore di 180 se si tratta di una triennale o 120 se si tratta di una magistrale.
  Uno studente può effettuare una prenotazione per un appello di seduta di laurea se e solo se ha pagato tutte le tasse previste, la data di prenotazione è avvenuta nel range di date corretto, se ci sono abbastanza posti rimanenti e se ha effettuato il tirocinio.
- Uno studente non può pagare la stessa tassa due volte.
- Uno studente non può effetuare la domanda di partecipazione a un bando Erasmus o a un bando di borsa di studio prima della data di emissione del bando o dopo la sua scadenza.
- Una borsa di studio di un determinato bando di borsa di studio può essere assegnata allo studente solo se questi ha effettuato una prenotazione per quel determinato bando di borsa di studio, se ci sono borse ancora disponibili, se non è stato assegnatario di borsa di studio nello stesso anno accademico. In maniera analoga per i bandi Erasmus.
- Uno studente può compilare il questionario di una determinata edizione di insegnamento se e solo se ha frequentato quella specifica edizione di insegnamento, se l'edizione insegnamento risulta essere relativa al corso di laurea al quale è iscritto lo studente, se il docente per il quale si compila il questionario insegna effettivamente quell'edizione di insegnamento, se non è stato già compilato.
- Uno studente può effettuare il tirocinio se e soltanto se ha completato determinati esami oppure se ha raggiunto una quota minima di CFU. In maniera analoga per l'assegnazione di un relatore a uno studente che ne fa richiesta.
- Alcuni appelli di determinate edizioni insegnamento possono essere riservati solo per studenti fuoricorso oppure per tutti gli studenti.
- Per poter sostenere gli esami di un certo anno di un corso, lo studente deve aver prima sostenuto gli esami dello stesso corso previsti per l'anno di corso precedente. Cioè studenti del secondo anno non possono sostenere esami del terzo anno, se non hanno già completato tutti gli esami del secondo anno previsti dal corso di studi.
- Uno studente non può partecipare a più seminari contemporaneamente.

Si precisa inoltre che non sono stati implementati tutti i vincoli dinamici appena riportati e che quelli riportati sono solo una piccola parte di quelli esistenti in un contesto reale.

Verifica di normalità

Nella verifica di normalità si combatte la ridondanza attraverso la caccia alle dipendenze funzionali anomale. Mentre il rispetto della prima forma normale è in qualche modo dato per scontato, implicito nel modello relazionale, la seconda e la terza vanno verificate analizzando il significato degli attributi, poiché si tratta di una proprietà intensionale, non estensionale.

Prima forma normale

La prima forma normale (1NF) è ora considerata parte integrante della definizione formale di relazione nel modello relazionale di base; storicamente è stata definita per non permettere l'uso di attributi multivalore, di attributi composti e delle loro combinazioni. Essa richiede che il dominio di un attributo comprenda solo valori atomici (semplici, indivisibili) e che il valore di qualsiasi attributo in una tupla sia un valore singolo del dominio di quell'attributo.

Il DB risulta essere in prima forma normale dal momento che non si fa uso di attributi NON strutturati, attributi strutturati ma NON decomposti nei rispettivi attributi atomici o attributi complessi NON atomici (più informazioni in un singolo attributo). Infatti, partiamo dalla questione della matricola degli studenti. La scelta è stata quella di non usare la costruzione ben nota per alcune università che consiste nel combinare il codice di corso di laurea di uno studente con uno o più simboli (come slash, backslash, trattino e così via) e con un contatore a tutti gli effetti. Nell'insieme tale codice risulta identificare univocamente uno studente. Si nota che in tal caso il contatore può assumere valori identici per corsi di laurea differenti. Nonostante ciò, non si creano problemi in quanto la chiave primaria è la combinazione del codice del corso di laurea con il valore contatore successivo. In tal caso però non sarebbe stata rispettata nemmeno la prima forma normale, perché il numero di matricola non è né un numero, né un valore atomico. Non è un numero perché è un codice e non è atomico perché è composto da un codice di corso di laurea e un codice studente.

Per risolvere tale problematica, si è pensato di utilizzare come chiave primaria dell'entità studente solo il contatore progressivo numerico, cioè quello che nell'approccio poc'anzi analizzato era la seconda parte della combinazione della chiave primaria. Così facendo però, non possono esservi più studenti con lo stesso "codice studente" per corsi di laurea differenti. Quindi si è ritenuto opportuno rendere univoco tale campo e tenere separate le informazioni relative al corso di laurea e quelle relative allo studente. Un approccio alternativo sarebbe stato quello di rendere l'entità studente debole rispetto al corso di laurea, combinando sia la separazione in valori atomici dei campi *corso di laurea* e *matricola studente* e raggiungendo la dipendenza di uno studente sia dal suo valore del codice studente che dal codice di corso di laurea al quale è iscritto. In ogni caso, si è preferito NON usare tale approccio ma quello di rendere forti sia l'entità *studente* che l'entità *corso di laurea.*

Il campo data è un tipo primitivo nella maggior parte dei linguaggi di programmazione, e pur essendo in realtà strutturato è convenzionalmente considerato atomico. Vale lo stesso per il campo data e ora di Oracle DBMS, a patto di compiere una forzatura più ampia dell'accezione di atomicità. Quindi in tal caso, i campi *data e ora* presenti non vanno ad intaccare la 1NF del DB. La conseguenza indesiderata di questa scelta è che la chiave delle entità deboli (che prevedono una data come parte integrante della combinazione di attributi realizzanti la chiave primaria), includa solo una parte dell'attributo *data e ora* e bisogna prevenire anomalie di inserimento.
Inoltre, per quanto riguarda attributi multivalore come *numero di telefono* ed *e-mail,* si conserva la 1NF grazie alla traduzione opportuna nel modello relazionale che prevede di creare una relazione per i *telefoni* e per le *e-mail* sia per gli *studenti*, sia per i *docenti*, sia per i *tutor aziendali.*

Discorso analogo per *l'orario delle lezioni* di un'edizione insegnamento. Mentre per quanto concerne *l'indirizzo*, questo viene suddiviso in più attributi atomici così da mantenere la 1NF. Quindi, il DB risulta essere in prima forma normale.

Seconda forma normale

La seconda forma normale (2NF) si basa sul concetto di dipendenza funzionale completa. Uno schema di relazione R è in 2NF se ogni attributo non-primo A di R dipende funzionalmente in modo completo dalla chiave primaria di R. Cioè, ogni attributo non-primo non deve avere una dipendenza parziale dalla PK (solo da una parte di essa).

Nel caso di questo DB, tutte le entità forti hanno una chiave primaria formata da un solo attributo. Per quanto riguarda le entità deboli e le relazioni "relazione", gli attributi non-primi in esse presenti dipendono funzionalmente dall'intera chiave non generando dipendenze parziali in nessun caso. Quindi date per buone le nozioni di atomicità definite nella discussione della 1NF, la seconda forma normale è anch'essa rispettata.

Terza forma normale

La terza forma normale (3NF) è basata sul concetto di dipendenza transitiva e sulle anomalie di aggiornamento. La relazione non deve contenere alcun attributo non-chiave funzionalmente dipendente da un altro attributo non-chiave (o da un insieme di attributi non-chiave). Cioè, non deve esistere alcuna dipendenza transitiva tra attributi non-chiave e la chiave primaria.

Nel caso di questo DB, per ogni relazione considerata la 3NF viene rispettata. Si pongono ora alcuni casi di studio affrontati durante la verifica della terza forma normale del DB esse4.

- Come accennato precedentemente, non vi è il codice fiscale né su *studente*, né su *docente*, né su *tutor aziendale.* Tale scelta è motivata dal fatto che la presenza del codice fiscale su una qualsiasi di queste tre tabelle, avrebbe comportato un campo non-chiave verso il quale ci sarebbero state delle dipendenze funzionali a partire da altri attributi. Ciò avrebbe violato la 3NF. Infatti, per come è stato modellato il DB, se uno studente cambiasse carriera o si iscrivesse a un nuovo corso di laurea triennale o scegliesse di proseguire gli studi con un corso di studi magistrale, si riproporrebbe lo stesso codice fiscale all'interno della relazione *studente*. Ciò risulta essere un problema, perché qualora si modellasse il contesto in maniera differente allora si potrebbe lasciare il codice fiscale e magari usarlo come chiave primaria in un'eventuale specializzazione di *persona.* Il fatto che però non abbia l'univocità fa sì che non possa essere considerato tra l'elenco delle possibili superchiavi minimali da scegliere come chiave primaria dell'entità. Quindi, visto e considerato che porterebbe a violare la 3NF, è stato eliminato completamente come attributo. Nel caso di *docente* e *tutor aziendale*, il discorso è analogo. Inoltre, qualora non cancellassimo il codice fiscale e lo usassimo come chiave esterna in un "tabellone" delle e-mail e in un "tabellone" dei telefoni (dove sono presenti indistintamente tutti i telefoni e tutte le e-mail sia di *studenti*, sia di *docenti*, che di *tutor aziendali*), avremmo molteplici difficoltà a tenere traccia dello studente, docente o tutor aziendale a cui rispettivamente fa riferimento quel numero di telefono o quella e-mail, dal momento che il codice fiscale non risulterebbe essere univoco. Quindi, si è scelto di creare tre tabelle separate per i numeri di telefono e tre tabelle separate per le e-mail.
- Si potrebbe pensare che il codice IUV sia un attributo non-chiave dal momento che è non obbligatorio. Però in base all'approfondita analisi dei requisiti che è stata effettuata per la realizzazione del seguente progetto, tale codice IUV viene comunque generato sia che il pagamento avvenga tramite il noto servizio di *pagoPA,* che senza (avviene indirettamente). Quindi risulta essere un attributo chiave e non c'è la violazione della 3NF.

- Si potrebbe pensare che il numero di CFU ottenuti mediante un tirocinio dipenda funzionalmente dalla combinazione di *data inizio* e *data fine* del *tirocinio* svolto dallo studente. In realtà, il numero di CFU è prestabilito a seconda del corso di laurea al quale lo studente è iscritto e i tre attributi non dipendono funzionalmente tra di loro in nessun modo. Quindi anche in questo caso, è rispettata la 3NF.

BCNF (Boyce-Codd Normal Form)

La BCNF risulta essere una forma normale molto più stringente della 3NF. Infatti, prevede che la relazione non deve contenere alcun attributo non-chiave funzionalmente dipendente da un altro attributo chiave o non-chiave (o da un insieme di attributi chiave o non-chiave). Cioè non vi possono essere nemmeno dipendenze funzionali verso attributi chiave, ma soltanto verso la chiave primaria della relazione.

Nel caso di questo DB, non risulta essere in BCNF. Infatti, a causa del codice IUV sull'entità tassa, del nome dell'azienda sull'entità azienda e altri casi simili, non è possibile determinare che il DB sia in BCNF. Tuttavia, la progettazione di un DB dovrebbe sforzarsi di raggiungere la 3NF o la BCNF. Forme normali più avanzate potrebbero portare a problemi di prestazione, perdite di dipendenze funzionali e quindi di vincoli di integrità. In alcuni casi, si applica addirittura la denormalizzazione. Quindi per concludere, il DB si trova in 3NF.

Possibili estensioni

Le estensioni possibili sono innumerevoli e il limite è dato solo dalla fantasia. Per esempio, si potrebbero gestire diversamente le eccezioni presenti evitando il fallimento dell'operazione e proseguendo in maniera alternativa. Magari presentando la vista delle prenotazioni che è possibile fare per quel determinato studente.

Altre estensioni potrebbero essere implementate con le opportune modifiche, estensioni e procedure:

- Storico migliori studenti
- Storico presidenti appelli per una determinata edizione insegnamento o per un determinato insegnamento
- Visualizzazione dati statistici come la media del voto degli studenti che hanno superato l'esame, la loro età, il periodo dell'anno in cui vengono promossi o bocciati più studenti o altre informazioni utili.
- Implementazione di una prenotazione a un appello subordinata all'eventuale consegna di un progetto. Per esempio, lo studente può prenotarsi a un appello di un'edizione di un insegnamento se e solo se ha consegnato il progetto entro una determinata data.
- Uno studente può chiedere ricevimento con un determinato docente per una specifica edizione dell'insegnamento solo un numero prefissato di volte al mese.
- Visualizzazione di dati statistici descrittivi dei più svariati fenomeni che si vogliono analizzare in questo contesto mediante il DB.
- Si può far sì che la procedura di programmazione appelli di una determinata edizione di insegnamento riesca a determinare le date di appello anche in base alle festività locali e nazionali.

Ulteriori possibili estensioni sono la gestione della banda, della memoria, gli aspetti relativi alla sicurezza, alla privatezza e alla concorrenza, le politiche di *backup*, l'ottimizzazione di performance. Per quanto concerne lo scheduler, ulteriori estensioni potrebbero consistere nell'andare a realizzare un job che ogni mese programma la data degli appelli di determinate edizione di insegnamento di un corso in maniera automatica e tale che non siano in discrepanza o accavallate con altre date di altri appelli di altre edizioni insegnamento dello stesso corso e dello stesso anno di corso. Così facendo si eviterebbero eventuali problemi che vanno contro il regolamento universitario.
Un'altra estensione dello scheduler consiste nell'andare a programmare i ricevimenti dei docenti ogni settimana con una determinata politica di turnazione per gli studenti a seconda dei motivi del ricevimento, oppure andando a introdurre politiche premiali o sanzionatorie per coloro che spesso disdicono un ricevimento poco prima che abbia luogo.
Un'altra estensione dello scheduler consiste nell'andare a programmare gli appelli di seduta di laurea definendo con dei metodi avanzati quali docenti ne faranno parte, oppure l'assegnazione del presidente per un appello di un'edizione insegnamento che prevede più docenti ad impartire quel determinato corso.

Implementazione

Utenti
DDL
DML
Trigger
Procedure
Viste
DCL
Scheduler

In questo capitolo si tratta l'implementazione del database. Tutto quanto documentato nel capitolo precedente è tradotto in codice eseguibile sul DBMS Oracle. In particolare, sono state utilizzate 2 versioni: quella on-line di Oracle Live SQL (19c) e la Oracle 18c Express Edition.

Implementazione

”We use the dark IDE theme when coding because light attracts bugs.“

Una volta terminata la fase di progettazione, si passa alla fase di implementazione. Ciò consiste nell'andare a convertire tutto quello che è stato detto finora in qualcosa di effettivamente funzionante sotto forma di codice. Il codice si riferisce al DBMS Oracle e il linguaggio adottato è il PL/SQL. L'ordine con cui lanciare gli script non è necessariamente quello con cui sono presentati nel capitolo, che riflette piuttosto l'organizzazione logica dell'argomento. Tutti i nomi arbitrari sono evidenziati in nero, mentre le parole riservate sono evidenziate in blu.

Creazione Utenti

Come prima cosa bisogna accedere al DBMS come amministratore di sistema e creare l'utente proprietario della base di dati. Contestualmente possono essere creati anche gli altri utenti necessari ma, non esistendo ancora lo schema, non possono ricevere alcun privilegio di oggetto.

```
CREATE USER c##db_esse4 IDENTIFIED BY dbes;
CREATE USER c##docente IDENTIFIED BY doce;
CREATE USER c##studente IDENTIFIED BY stud;
CREATE USER c##segreteria IDENTIFIED BY segr;

GRANT ALL PRIVILEGES TO c##db_esse4;

GRANT CREATE ANY VIEW TO c##db_esse4;
GRANT CREATE ANY VIEW TO c##segreteria;

GRANT CREATE ANY PROCEDURE TO c##db_esse4;
GRANT CREATE ANY PROCEDURE TO c##segreteria;
```

Successivamente è necessario disconnettersi e connettersi nuovamente, questa volta però con le credenziali dell'utente amministratore del DB, *esse4*. In questo modo si avranno i permessi per creare gli oggetti necessari al database. Si precisa che all'amministratore vengono garantiti tutti i privilegi, mentre alla segreteria una porzione consistente di essi.

Data Definition Language

Innanzitutto, va sottolineato che in Oracle il tipo DATE include l'ora, e dunque i campi di tipo data e ora (con tutte le loro varianti come i campi ora e minuto, minuto e secondo etc.), pur se separati nel diagramma E/R, sono tradotti con un campo unico di tipo DATE. Il DDL è la traduzione del modello relazionale in linguaggio SQL. Alcuni dettagli del relazionale però, non vengono espressi per evitare loop e/o problemi di popolamento, oppure perché sono insiti nella definizione di particolari keyword di sistema, oppure per altri motivi. Useremo le istruzioni CREATE TABLE per andare a creare le tabelle necessarie e nel frattempo esprimeremo i vari vincoli esprimibili nel relazionale come l'obbligatorietà, i vincoli di dominio statico, l'univocità, l'integrità referenziale, etc. In generale, si noti come la totalità della FK sia espressa implicitamente dal vincolo di PK in alcune tabelle.

Corso Laurea

```
create table corso_laurea (
  codice_corso char(15) primary key,
  tipo varchar2(15) not null,
  capienza number not null,
  nome varchar2(60) not null,
  constraint check_tipo1 check (
    tipo in (
      'MAGISTRALE', 'TRIENNALE', 'magistrale', 'triennale', 'Magistrale', 'Triennale'
    )
  ),
  constraint check_capienza1 check (capienza > 0)
);
```

Studente

```
-- totalità rispetto a Corso di Laurea espressa con una NOT NULL sulla relativa FK
create table studente (
  matricola_studente char(15) primary key,
  nome varchar2(30) not null,
  cognome varchar2(30) not null,
  data_nascita date not null,
  sesso char(1) default 'U',
  via varchar2(50),
  numero_civico varchar2(10),
  CAP char(5),
  citta varchar2(30),
  codice_corso char(15) not null,
  data_iscrizione date not null,
  constraint check_sesso1 check (sesso in ('M', 'F', 'U', 'm', 'f', 'u')),
  constraint fk_codice_corso1 foreign key (codice_corso) references
       corso_laurea(codice_corso) on delete cascade,
  constraint check_date13 check (data_nascita < data_iscrizione)
);
```

Docente

```
create table docente (
  numero_tesserino char(15) primary key,
  nome varchar2(30) not null,
  cognome varchar2(30) not null,
  data_nascita date not null,
  sesso char(1) default 'U',
  via varchar2(50),
  numero_civico varchar(10),
  CAP char(5),
  citta varchar2(30),
  constraint check_sesso2 check (sesso in ('M', 'F', 'U', 'm', 'f', 'u'))
);
```

Insegnamento

```
create table insegnamento (
  codice_insegnamento char(15) primary key,
  nome varchar2(60) not null
);
```

Edizione insegnamento

```
create table edizione_insegnamento (
  codice_insegnamento char(15),
  anno_accademico date,
  semestre varchar2(15) not null,
  anno_corso number(1, 0) not null,
  CFU number(2, 0) not null,
  svolgimento varchar2(15) not null,
  constraint pk_edizione_insegnamento primary key (codice_insegnamento, anno_accademico),
  constraint check_semestre1 check (
    semestre in (
      'PRIMO', 'SECONDO', 'ANNUALE', 'Primo', 'Secondo', 'Annuale', 'primo', 'secondo',
      'annuale')),
  constraint check_anno_corso1 check (anno_corso in (1, 2, 3, 4, 5)),
  constraint check_svolgimento check (
    svolgimento in (
      'Presenza', 'Telematica', 'presenza', 'telematica', 'PRESENZA', 'TELEMATICA')
    ),
  constraint check_CFU3 check(CFU > 0),
  constraint check_CFU4 check(CFU < 13),
  constraint fk_codice_insegnamento2 foreign key (codice_insegnamento) references
       insegnamento(codice_insegnamento) on delete cascade
);
```

Offerta insegnamento

```
create table offerta_insegnamento (
  codice_insegnamento char(15),
  codice_corso char(15),
  corso_principale char(1) not null,
  constraint pk_offerta_insegnamento primary key (codice_insegnamento, codice_corso),
  constraint check_corso_principale1 check(corso_principale in ('Y', 'y', 'N', 'n')),
  constraint fk_codice_insegnamento1 foreign key (codice_insegnamento) references
       insegnamento(codice_insegnamento) on delete cascade,
  constraint fk_codice_corso2 foreign key (codice_corso) references
       corso_laurea(codice_corso) on delete cascade
);
```

Insegna edizione

```
create table insegna_edizione (
  codice_insegnamento char(15),
  anno_accademico date,
  tesserino_docente char(15),
  tipo_docente varchar2(30) not null,
  constraint pk_insegna_edizione primary key (
       codice_insegnamento, anno_accademico, tesserino_docente),
  constraint check_tipo_docente1 check (
    tipo_docente in ('TEORIA', 'Teoria', 'teoria', 'LABORATORIO', 'Laboratorio',
       'laboratorio', 'TEORIA E LABORATORIO', 'Teoria e Laboratorio', 'teoria e
       laboratorio')),
  constraint fk_tesserino_docente1 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete cascade,
  constraint fk_codice_insegnamento_anno_accademico1 foreign key (
    codice_insegnamento, anno_accademico) references edizione_insegnamento(
       codice_insegnamento, anno_accademico) on delete cascade
);
```

Frequenta edizione insegnamento

```
create table frequenta_edizione_insegnamento (
  anno_accademico date,
  codice_insegnamento char(15),
  matricola_studente char(15),
  data_ins date not null,
  constraint pk_frequenta_edizione_insegnamento primary key (
    anno_accademico, codice_insegnamento, matricola_studente),
  constraint fk_matricola_studente7 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint fk_codice_insegnamento_anno_accademico2 foreign key (
    codice_insegnamento, anno_accademico) references edizione_insegnamento(
       codice_insegnamento, anno_accademico) on delete cascade
);
```

Appello laurea

```
-- totalità rispetto a corso di laurea espressa con la NOT NULL sulla relativa FK
create table appello_laurea (
  data_appello date,
  codice_corso char(15) not null,
  inizio_iscrizioni date not null,
  fine_iscrizioni date not null,
  tipo varchar2(15) default 'Non previsto',
  max_iscrizioni number(2, 0) not null,
  constraint pk_appello_laurea primary key (data_appello, codice_corso),
  constraint check_max_iscrizioni1 check (max_iscrizioni > 0),
  constraint check_max_iscrizioni2 check (max_iscrizioni <= 20),
  constraint check_date2 check (inizio_iscrizioni < fine_iscrizioni),

  constraint check_tipo2 check (
    tipo in (
      'Presenza', 'Telematica', 'PRESENZA', 'TELEMATICA', 'presenza', 'telematica', 'Non
       previsto', 'NON PREVISTO', 'non previsto')),
  constraint check_date3 check (data_appello > fine_iscrizioni),
  constraint check_date4 check (inizio_iscrizioni < data_appello),
  constraint check_date12 check (inizio_iscrizioni < fine_iscrizioni),
  constraint fk_codice_corso3 foreign key (codice_corso) references
       corso_laurea(codice_corso) on delete cascade
);
```

Prenotazione appello seduta

```
create table prenotazione_appello_seduta (
  matricola_studente char(15),
  codice_corso char(15),
  data_appello date not null,
  data_prenotazione date not null,
  numero char(15) not null unique,
  constraint pk_prenotazione_appello_seduta primary key (
    matricola_studente, data_appello, codice_corso),
  constraint check_date5 check (data_prenotazione < data_appello),
  constraint fk_matricola_studente9 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint fk_codice_corso_data_appello_laurea1 foreign key (codice_corso, data_appello)
       references appello_laurea (codice_corso, data_appello) on delete cascade
);
```

Partecipa seduta

```
create table partecipa_seduta (
  tesserino_docente char(15),
  data_appello date,
  codice_corso char(15),
  presidente char(1) not null,
  constraint pk_partecipa_seduta primary key (
    tesserino_docente, data_appello, codice_corso),
  constraint check_presidente2 check (presidente in ('y', 'n', 'Y', 'N')),
  constraint fk_tesserino_docente6 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete cascade,
  constraint fk_codice_corso_data_appello_laurea2 foreign key (codice_corso, data_appello)
       references appello_laurea (codice_corso, data_appello) on delete cascade
);
```

Seduta laurea

```
-- totalità espressa rispetto a studente espressa con NOT NULL sulla relativa FK
create table seduta_laurea (
  numero_verbale char(15) primary key,
  codice_corso char(15),
  data_seduta date,
  matricola_studente char(15) unique not null,
  voto number(3, 0) not null,
  lode char(1) default 'N',
  constraint check_voto1 check (voto > 65),
  constraint check_voto2 check (voto <= 110),
  constraint check_lode1 check (
    (lode in ('N', 'n') OR (lode in ('Y', 'y') AND (voto = 110)))),
  constraint fk_codice_corso_data_appello_laurea3 foreign key (codice_corso, data_seduta)
       references appello_laurea (codice_corso, data_appello) on delete set null,
  constraint fk_matricola_studente4 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade
);
```

Relatore

```
create table relatore (
  matricola_studente char(15),
  tesserino_docente char(15),
  data_inizio date not null,
  data_fine date not null,
  titolo_tesi varchar2(100),
  tipo_tesi varchar2(15),
  constraint pk_relatore primary key (matricola_studente, tesserino_docente),
  constraint check_tipo_tesi1 check (
    tipo_tesi in (
      'Compilativa', 'COMPILATIVA', 'compilativa',
      'Sperimentale', 'SPERIMENTALE', 'sperimentale')),
  constraint fk_matricola_studente2 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint fk_tesserino_docente2 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete set null,
  constraint check_date1 check (data_inizio < data_fine)
);
```

Azienda

```
create table azienda (
  partita_iva char(15) primary key,
  nome varchar2(30) not null,
  via varchar2(20),
  numero_civico varchar2(10),
  CAP char(5),
  citta varchar2(30) not null
);
```

Tutor aziendale

```
-- totalità rispetto ad azienda espressa con la NOT NULL sulla relativa FK
create table tutor_aziendale (
  numero_tesserino char(15) primary key,
  nome varchar2(30) not null,
  cognome varchar2(30) not null,
  data_nascita date not null,
  sesso char(1) default 'U',
  via varchar2(50),
  numero_civico varchar2(10),
  CAP char(5),
  citta varchar2(30),
  partita_iva char(15) not null,
  constraint check_sesso3 check (sesso in ('M', 'F', 'U', 'm', 'f', 'u')),
  constraint fk_partita_iva1 foreign key (partita_iva) references azienda(partita_iva) on
       delete cascade
);
```

Tirocinio

```
-- totalità rispetto a studente e docente espressa, ma non rispetto a tutor aziendale
-- per come sono impostate le totalità la politica di reazione più appropriata è la ON
DELETE CASCADE su studente e sulle altre ON DELETE SET NULL
create table tirocinio (
  numero_tirocinio char(15) primary key,
  CFU number(2, 0) not null,
  data_inizio date not null,
  data_fine date not null,
  tesserino_docente char(15) not null,
  tesserino_tutor_azienda char(15),
  matricola_studente char(15) not null unique,
  constraint fk_tesserino_docente4 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete set null,
  constraint fk_matricola_studente5 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint fk_tesserino_tutor_aziendale1 foreign key (tesserino_tutor_azienda) references
       tutor_aziendale(numero_tesserino) on delete set null,
  constraint check_date7 check (data_inizio < data_fine),
  constraint check_CFU1 check (CFU > 0),
  constraint check_CFU2 check (CFU < 13)
);
```

Questionario

```
-- totalità rispetto a docente e a studente espressa con NOT NULL
create table questionario (
  numero_questionario char(15),
  codice_insegnamento char(15),
  anno_accademico date,
  materiale_didattico number(1, 0) not null,
  gradimento number(1, 0) not null,
  disponibilita_docente number(1, 0) not null,
  precisione_orario number(1, 0) not null,
  tesserino_docente char(15) not null,
  matricola_studente char(15) not null,
  data_compilazione date not null,
  constraint pk_questionario primary key (
    numero_questionario, codice_insegnamento, anno_accademico),
  constraint check_materiale_didattico1 check (materiale_didattico > 0),
  constraint check_materiale_didattico2 check (materiale_didattico <= 5),
  constraint check_gradimento1 check (gradimento > 0),
  constraint check_gradimento2 check (gradimento <= 5),
  constraint check_disponibilita_docente1 check (disponibilita_docente > 0),
  constraint check_disponibilita_docente2 check (disponibilita_docente <= 5),
  constraint check_precisione_orario1 check (precisione_orario > 0),
  constraint check_precisione_orario2 check (precisione_orario <= 5),
```

```
  constraint fk_codice_insegnamento_anno_accademico3 foreign key (
    codice_insegnamento, anno_accademico) references edizione_insegnamento(
    codice_insegnamento, anno_accademico) on delete cascade,
  constraint fk_matricola_studente18 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint fk_tesserino_docente10 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete cascade
);
```

Appello

```
create table appello (
  anno_accademico date,
  data_appello date,
  codice_insegnamento char(15),
  data_inizio date not null,
  data_fine date not null,
  max_studenti number(3, 0) not null,
  tipo varchar2(15) default 'Non previsto',
  constraint pk_appello primary key (anno_accademico, data_appello, codice_insegnamento),
  constraint check_max_studenti1 check (max_studenti > 0),
  constraint check_tipo3 check (
    tipo in (
      'Orale', 'ORALE', 'orale', 'scritto', 'Scritto', 'SCRITTO', 'Non previsto',
      'NON PREVISTO', 'non previsto')),
  constraint check_date6 check (data_inizio < data_fine),
  constraint check_date10 check (data_inizio < data_appello),
  constraint check_date11 check (data_fine < data_appello),
  constraint fk_codice_insegnamento_anno_accademico4 foreign key (
    codice_insegnamento, anno_accademico) references edizione_insegnamento(
    codice_insegnamento, anno_accademico) on delete cascade
);
```

Orario lezioni

```
create table orario_lezioni (
  giorno_e_ora date,
  anno_accademico date,
  codice_insegnamento char(15),
  constraint pk_orario_lezioni primary key (
    giorno_e_ora, anno_accademico, codice_insegnamento),
  constraint fk_codice_insegnamento_anno_accademico5 foreign key (
    codice_insegnamento, anno_accademico) references edizione_insegnamento(
    codice_insegnamento, anno_accademico) on delete cascade
);
```

Seminario

```
create table seminario (
  data_seminario date,
  tesserino_docente char(15),
  nome varchar2(30),
  CFU number(2, 0) not null,
  max_persone number(3, 0) not null,
  constraint pk_seminario primary key (data_seminario, tesserino_docente),
  constraint check_max_persone1 check (max_persone > 0),
  constraint fk_tesserino_docente5 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete set null,
  constraint check_CFU6 check (CFU >= 0)
);
```

Presiede appello

```
create table presiede_appello (
  anno_accademico date,
  data_appello date,
  codice_insegnamento char(15),
  tesserino_docente char(15),
  presidente char(1) not null,
  constraint pk_presiede_appello primary key (
    anno_accademico, data_appello, codice_insegnamento, tesserino_docente),
  constraint check_presidente1 check (presidente in ('y', 'n', 'Y', 'N')),
  constraint fk_tesserino_docente13 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete cascade,
  constraint fk_codice_insegnamento_anno_accademico_data_appello1 foreign key (
    codice_insegnamento, anno_accademico, data_appello) references appello(
    codice_insegnamento, anno_accademico, data_appello) on delete cascade
);
```

Ricevimento

```
create table ricevimento (
  data_ricevimento date,
  tesserino_docente char(15),
  durata number(3, 0) default 0,
  constraint pk_ricevimento primary key (data_ricevimento, tesserino_docente),
  constraint fk_tesserino_docente15 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete cascade
);
```

Esame superato

```
-- totalità rispetto a studente espressa con la NOT NULL
-- totalità rispetto ad appello non può essere espressa
create table esame_superato (
  numero_verbale char(15) primary key,
  voto number(2) not null,
  lode char(1) not null,
  data_esame date,
  anno_accademico date,
  codice_insegnamento char(15),
  matricola_studente char(15) not null,
  constraint check_voto3 check (voto > 17),
  constraint check_voto4 check (voto <= 30),
  constraint check_lode2 check (
    (lode in ('N', 'n') OR (lode in ('Y', 'y') AND (voto = 30)))),
  constraint check_univocita_studente unique(matricola_studente, codice_insegnamento),
  constraint fk_codice_insegnamento_anno_accademico_data_appello2 foreign key (
    codice_insegnamento, anno_accademico, data_esame) references appello(
    codice_insegnamento, anno_accademico, data_appello) on delete set null,
  constraint fk_matricola_studente13 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade
);
```

Prenotazione ricevimento

```
create table prenotazione_ricevimento (
  numero_prenotazione char(15) not null unique,
  data_ricevimento date,
  tesserino_docente char(15),
  matricola_studente char(15),
  data_prenotazione date not null,
  constraint pk_prenotazione_ricevimento primary key (
    data_ricevimento, tesserino_docente, matricola_studente),
  constraint fk_data_ricevimento_docente1 foreign key (
    data_ricevimento, tesserino_docente) references ricevimento(
    data_ricevimento, tesserino_docente) on delete cascade,
  constraint fk_matricola_studente19 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
```

```
  constraint check_date8 check (data_prenotazione < data_ricevimento));
```

Prenotazione appello

```
create table prenotazione_appello (
  anno_accademico date,
  data_appello date,
  codice_insegnamento char(15),
  matricola_studente char(15),
  numero_prenotazione char(15) not null,
  data_prenotazione date not null,
  constraint pk_prenotazione_appello primary key (
    anno_accademico, data_appello, codice_insegnamento, matricola_studente),
  constraint fk_matricola_studente12 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint fk_codice_insegnamento_anno_accademico_data_appello3 foreign key (
    codice_insegnamento, anno_accademico, data_appello) references appello(
    codice_insegnamento, anno_accademico, data_appello) on delete cascade,
  constraint check_date9 check (data_prenotazione < data_appello)
);
```

Tassa

```
-- totalità rispetto a studente espressa a differenza di quanto riportato nel relazionale
create table tassa (
  numero_fattura char(15) primary key,
  scadenza date not null,
  importo number(3, 0) not null,
  matricola_studente char(15) not null,
  data_pagamento date,
  mora number(3, 0) default 0,
  IUV char(25),
  tipo varchar2(15),
  constraint check_importo1 check (importo > 0),
  constraint fk_matricola_studente15 foreign key (matricola_studente) references
studente(matricola_studente) on delete set null
);
```

Partecipa seminario

```
-- In questo caso la politica di reazione più appropriata è on delete cascade se viene
-- cancellato uno studente e on delete se null se viene cancellato il seminario
create table partecipa_seminario (
  data_seminario date,
  tesserino_docente char(15),
  matricola_studente char(15),
  constraint pk_partecipa_seminario primary key (
    data_seminario, tesserino_docente, matricola_studente),
  constraint fk_matricola_studente20 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint fk_data_seminario_tesserino_docente1 foreign key (
    data_seminario, tesserino_docente) references seminario(
    data_seminario, tesserino_docente) on delete set null
);
```

Bando borsa

```
create table bando_borsa (
  numero_bando_borsa char(15) primary key,
  scadenza date not null,
  data_emissione date not null,
  valore number(4, 0) not null,
  causale varchar2(25),
  tipo varchar2(25),
  numero_borse number(4, 0) not null,
  constraint check_numero_borse1 check (numero_borse > 0),
  constraint check_valore1 check (valore > 0),
```

```
  constraint check_date14 check (data_emissione < scadenza)
);
```

Bando Erasmus

```
create table bando_erasmus (
  numero_bando_erasmus char(15) primary key,
  scadenza date not null,
  data_emissione date not null,
  CFU number(2, 0) not null,
  numero_posti number(3, 0) not null,
  constraint check_numero_posti1 check (numero_posti > 0),
  constraint check_CFU5 check (CFU > 0),
  constraint check_date15 check (data_emissione < scadenza)
);
```

Partecipazione bando borsa

```
create table partecipazione_bando_borsa (
  numero_bando_borsa char(15),
  matricola_studente char(15),
  data_domanda date not null,
  constraint pk_partecipazione_bando_borsa primary key (
    numero_bando_borsa, matricola_studente
  ),
  constraint fk_matricola_studente8 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint fk_numero_bando_borsa1 foreign key (numero_bando_borsa) references
       bando_borsa(numero_bando_borsa) on delete cascade
);
```

Partecipazione bando Erasmus

```
create table partecipazione_bando_erasmus (
  numero_bando_erasmus char(15),
  matricola_studente char(15),
  data_domanda date not null,
  constraint pk_partecipazione_bando_erasmus primary key (
    numero_bando_erasmus, matricola_studente),
  constraint fk_matricola_studente10 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint fk_numero_bando_erasmus1 foreign key (numero_bando_erasmus) references
       bando_erasmus(numero_bando_erasmus) on delete set null
);
```

Assegnazione Erasmus

```
create table assegnazione_erasmus (
  numero_bando_erasmus char(15),
  matricola_studente char(15),
  data_assegnazione date not null,
  localita varchar2(30),
  nome_universita varchar2(25),
  data_partenza date not null,
  data_rientro date not null,
  constraint pk_assegnazione_bando_erasmus primary key (
    numero_bando_erasmus, matricola_studente),
  constraint fk_numero_bando_erasmus2 foreign key (numero_bando_erasmus) references
       bando_erasmus(numero_bando_erasmus) on delete set null,
  constraint fk_matricola_studente17 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint check_date16 check (data_partenza < data_rientro),
  constraint check_date17 check (data_assegnazione < data_partenza),
  constraint check_date18 check (data_assegnazione < data_rientro)
);
```

Assegnazione borse

```
create table assegnazione_borse (
  numero_bando_borsa char(15),
  matricola_studente char(15),
  data_assegnazione date not null,
  constraint pk_assegnazione_bando_borsa primary key (
    numero_bando_borsa, matricola_studente),
  constraint fk_numero_bando_borsa2 foreign key (numero_bando_borsa) references
       bando_borsa(numero_bando_borsa) on delete cascade,
  constraint fk_matricola_studente16 foreign key (matricola_studente) references
       studente(matricola_studente) on delete set null
);
```

Telefono studente

```
create table telefono_studente (
  numero_telefono_studente char(10),
  matricola_studente char(15),
  constraint pk_telefono_studente primary key (
    numero_telefono_studente, matricola_studente),
  constraint fk_matricola_studente21 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade
);
```

Telefono docente

```
create table telefono_docente (
  numero_telefono_docente char(10),
  tesserino_docente char(15),
  constraint pk_telefono_docente primary key (numero_telefono_docente, tesserino_docente),
  constraint fk_tesserino_docente12 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete cascade
);
```

Telefono tutor aziendale

```
create table telefono_tutor_aziendale (
  numero_telefono_tutor_aziendale char(10),
  tesserino_tutor_azienda char(15),
  constraint pk_telefono_tutor_aziendale primary key (
    numero_telefono_tutor_aziendale, tesserino_tutor_azienda),
  constraint fk_tesserino_tutor_aziendale2 foreign key (tesserino_tutor_azienda) references
       tutor_aziendale(numero_tesserino) on delete cascade
);
```

Email studente

```
create table email_studente (
  mail_studente varchar2(50),
  matricola_studente char(15),
  constraint pk_email_studente primary key (mail_studente, matricola_studente),
  constraint fk_matricola_studente22 foreign key (matricola_studente) references
       studente(matricola_studente) on delete cascade,
  constraint email_studente check (REGEXP_LIKE(mail_studente,
       '^[A-Za-z]+\.[A-Za-z]+[0-9]+@studenti.uniparthenope.it$'))
);
```

Email docente

```
create table email_docente (
  mail_docente varchar2(50),
  tesserino_docente char(15),
  constraint pk_email_docente primary key (mail_docente, tesserino_docente),
  constraint fk_tesserino_docente14 foreign key (tesserino_docente) references
       docente(numero_tesserino) on delete cascade,
  constraint email_docente check (REGEXP_LIKE(mail_docente,
       '^[A-Za-z]+\.[A-Za-z]+@uniparthenope.it$')));
```

Email tutor aziendale

```
create table email_tutor_aziendale (
  mail_tutor_aziendale varchar2(50),
  tesserino_tutor_azienda char(15),
  constraint pk_email_tutor_aziendale primary key (
    mail_tutor_aziendale, tesserino_tutor_azienda),
  constraint fk_tesserino_tutor_aziendale3 foreign key (tesserino_tutor_azienda) references
       tutor_aziendale(numero_tesserino) on delete cascade,
  constraint email_tutor_aziendale check (REGEXP_LIKE(mail_tutor_aziendale,
       '^[A-Za-z]+\.[A-Za-z]+@uniparthenope.it$'))
);
```

Data Manipulation Language

Per eseguire il popolamento (così come nel caso della creazione delle tabelle) del DB *esse4*, è stato necessario seguire un particolare ordine dettato dall'organizzazione logica del DB. Tutte le tabelle vengono popolate mediante istruzioni di *INSERT* accuratamente codificate.

Corso Laurea

```
INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0124', 'TRIENNALE', 180, 'INFORMATICA');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0332', 'TRIENNALE', 180, 'INGEGNERIA CIVILE E AMBIENTALE PER LA MITIGAZIONE DEI RISCHI');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0328', 'TRIENNALE', 300, 'INGEGNERIA GESTIONALE');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0327', 'TRIENNALE', 300, 'INGEGNERIA INFORMATICA, BIOMEDICA E DELLE TELECOMUNICAZIONI');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0123', 'TRIENNALE', 180, 'SCIENZE BIOLOGICHE');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0512', 'TRIENNALE', 180, 'SCIENZE MOTORIE');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0122', 'TRIENNALE', 300, 'SCIENZE NAUTICHE, AERONAUTICHE E METEO-OCEANOGRAFICHE');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0126', 'MAGISTRALE', 300, 'BIOLOGIA PER LA SOSTENIBILITA');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0120', 'MAGISTRALE', 180, 'INFORMATICA APPLICATA(MACHINE LEARNING E BIG DATA');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0331', 'MAGISTRALE', 300, 'INGEGNERIA CIVILE E PER LA TUTELA DELL''AMBIENTE COSTIERO');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0330', 'MAGISTRALE', 300, 'INGEGNERIA DELLA SICUREZZA DEI DATI E DELLE COMUNICAZIONI');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0326', 'MAGISTRALE', 300, 'INGEGNERIA GESTIONALE');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0515', 'MAGISTRALE', 180, 'SCIENZE E MANAGEMENT DELLO SPORT E DELLE ATTIVITA''MOTORIE');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0121', 'MAGISTRALE', 180, 'SCIENZE E TECNOLOGIE DELLA NAVIGAZIONE');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0514', 'MAGISTRALE', 180, 'SCIENZE MOTORIE PER LA PREVENZIONE ED IL BENESSERE');

INSERT INTO CORSO_LAUREA(CODICE_CORSO, TIPO, CAPIENZA, NOME) VALUES
('0125', 'TRIENNALE', 180, 'CONDUZIONE DEL MEZZO NAVALE');
```

Insegnamento

```
INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('SNN', 'SICUREZZA DELLA NAVE E DELLA NAVIGAZIONE');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('ASD', 'ALGORITMI E STRUTTURE DATI E LABORATORIO');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('BD', 'BASI DI DATI E LABORATORIO');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('BIOC', 'BIOCHIMICA E LABORATORIO');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('CHIMG', 'CHIMICA GENERALE E LABORATORIO');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('OCEA', 'OCEANOGRAFIA');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('METEO', 'METEOROLOGIA');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('MACL', 'MACHINE LEARNING');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('COMG', 'COMPUTER GRAPHICS');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('BIODIV', 'BIODIVERSITA E INDICATORI BIOLOGICI');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('METEOA', 'METEOROLOGIA AVANZATA');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('NAVSAT', 'NAVIGAZIONE SATELLITARE');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('MICROB', 'MICROBIOLOGIA');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('SCIES', 'SCIENZA DEL SUOLO');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('IMPE', 'IMPIANTI ELETTRICI');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('MACC', 'MACCHINE');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('IDRA', 'IDRAULICA');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('CIDRA', 'COSTRUZIONI IDRAULICHE');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('PROGCE', 'PROGRAMMAZIONE DEI CALCOLATORI ELETTRONICI');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('ELETT', 'ELETTRONICA');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('OCEAC', 'OCEANOGRAFIA COSTIERA');
```

```
INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('ENED', 'ENERGETICA DEGLI EDIFICI');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('RTI', 'RETI DI TELECOMUNICAZIONI E INTERNET');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('TI', 'TEORIA DELL''INFORMAZIONE');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('GSE', 'GESTIONE DEI SISTEMI ENERGETICI');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('AI', 'AUTOMAZIONE INDUSTRIALE');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('END', 'ENDOCRINOLOGIA');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('NEU', 'NEUROLOGIA');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('AEA', 'AGONISMO ED EDUCAZIONE IN ADOLESCENZA');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('BPAS', 'BUSINESS PLAN DELLE AZIENDE SPORTIVE');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('FISGE', 'FISIOPATOLOGIA GENERALE APPLICATA ALLE SCIENZE MOTORIE');

INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES
('MMSD', 'MECCANISMI MOLECOLARI DELLE SOSTANZE DOPANTI');
```

Offerta Insegnamento

```
INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('ASD', '0124', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('BD', '0124', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('METEO', '0125', 'N');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('SNN', '0125', 'N');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('BIOC', '0123', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('CHIMG', '0123', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('METEO', '0122', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('OCEA', '0122', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('MACL', '0120', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('COMG', '0120', 'Y');
```

```
INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('BIODIV', '0126', 'Y') ;

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('METEOA', '0121', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('NAVSAT', '0121', 'N');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('MICROB', '0126', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('SCIES', '0126', 'N');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('IMPE', '0328', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('MACC', '0328', 'N');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('IDRA', '0332', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('CIDRA', '0332', 'N');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('PROGCE', '0326', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('ELETT', '0326', 'N');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('OCEAC', '0331', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('ENED', '0331', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('RTI', '0330', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('TI', '0330', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('GSE', '0332', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('AI', '0332', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('END', '0512', 'N');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('NEU', '0512', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('AEA', '0515', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('BPAS', '0515', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('FISGE', '0514', 'Y');
```

```
INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('MMSD', '0514', 'Y');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('ASD', '0327', 'N');

INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
VALUES ('BIOC', '0327', 'N');
```

Edizione Insegnamento

```
INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('ASD', TO_DATE('2019','YYYY'), 'PRIMO', 1, 12, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('SNN', TO_DATE('2019','YYYY'), 'PRIMO', 1, 12, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('BD', TO_DATE('2019','YYYY'), 'SECONDO', 1,  9, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('BIOC', TO_DATE('2019','YYYY'), 'PRIMO', 1, 12, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('CHIMG', TO_DATE('2019','YYYY'), 'PRIMO', 1, 9, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('OCEA', TO_DATE('2019','YYYY'), 'SECONDO', 1, 12, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('METEO', TO_DATE('2019','YYYY'), 'PRIMO', 1, 9, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('MACL', TO_DATE('2019','YYYY'), 'PRIMO', 1, 12, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('COMG', TO_DATE('2019','YYYY'), 'SECONDO', 1, 9, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('BIODIV', TO_DATE('2019','YYYY'), 'SECONDO', 1, 9, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('METEOA', TO_DATE('2019','YYYY'), 'SECONDO', 1, 9, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('NAVSAT', TO_DATE('2019','YYYY'), 'PRIMO', 1, 6, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('MICROB', TO_DATE('2019','YYYY'), 'PRIMO', 1, 12, 'TELEMATICA');
```

```
INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('SCIES', TO_DATE('2019','YYYY'), 'SECONDO', 1, 12, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('IMPE', TO_DATE('2019','YYYY'), 'SECONDO', 1, 9, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('MACC', TO_DATE('2019','YYYY'), 'SECONDO', 1, 6, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('IDRA', TO_DATE('2019','YYYY'), 'PRIMO', 1, 9, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('CIDRA', TO_DATE('2019','YYYY'), 'SECONDO', 1, 12, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('PROGCE', TO_DATE('2019','YYYY'), 'PRIMO', 1, 6, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('ELETT', TO_DATE('2019','YYYY'), 'SECONDO', 1, 9, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('OCEAC', TO_DATE('2019','YYYY'), 'PRIMO', 1, 9, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('ENED', TO_DATE('2019','YYYY'), 'PRIMO', 1, 12, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('RTI', TO_DATE('2019','YYYY'), 'PRIMO', 1, 3, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('TI', TO_DATE('2019','YYYY'), 'PRIMO', 1, 3, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('GSE', TO_DATE('2019','YYYY'), 'SECONDO', 1, 9, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('AI', TO_DATE('2019','YYYY'), 'PRIMO', 1, 12, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('END', TO_DATE('2019','YYYY'), 'PRIMO', 1, 6, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('NEU', TO_DATE('2019','YYYY'), 'SECONDO', 1, 9, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('AEA', TO_DATE('2019','YYYY'), 'PRIMO', 1, 3, 'PRESENZA');
```

```
INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('BPAS', TO_DATE('2019','YYYY'), 'PRIMO', 1, 12, 'TELEMATICA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('FISGE', TO_DATE('2019','YYYY'), 'PRIMO', 1, 12, 'PRESENZA');

INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
('MMSD', TO_DATE('2019','YYYY'), 'SECONDO', 1, 9, 'TELEMATICA');
```

Orario Lezioni

```
INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'SNN');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'ASD');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'BD');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'BIOC');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'), 'CHIMG');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'), TO_DATE('2019','YYYY'),'OCEA');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'METEO');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'), 'MACL');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'COMG');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'BIODIV');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('11/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'METEOA');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'NAVSAT');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'), 'MICROB');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'SCIES');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'IMPE');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'MACC');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'IDRA');
```

```
INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'CIDRA');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'PROGCE');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'ELETT');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'), 'OCEAC');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'ENED');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'RTI');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'TI');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'GSE');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'AI');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'END');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'NEU');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'), 'AEA');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'BPAS');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('10/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'),'FISGE');

INSERT INTO ORARIO_LEZIONI(GIORNO_E_ORA, ANNO_ACCADEMICO, CODICE_INSEGNAMENTO) VALUES
(TO_DATE('01/10/2019 10:00:00', 'DD/MM/YYYY hh24:mi:ss'),TO_DATE('2019','YYYY'), 'MMSD');
```

Docente

```
INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111111','MASSIMILIANO','ROSSI',TO_DATE('11/05/1971','DD/MM/YYYY'),'M','VIA
MAZZINI','2','80100','NAPOLI');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111112','GIORGIO','BIANCHI',TO_DATE('19/01/1961','DD/MM/YYYY'),'M','VIA
GENOVA','15','80100','NAPOLI');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111113','MARIO','VERDI',TO_DATE('22/08/1975','DD/MM/YYYY'),'M','VIA
NAPOLEONE','33','80500','SALERNO');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111114','MARIAROSARIA','CASIMENI',TO_DATE('29/03/1974','DD/MM/YYYY'),'F','VIA DEI
LAMPIONI','2','80300','AVELLINO');
```

```
INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111115','MARIA','CAVINO',TO_DATE('11/08/1969','DD/MM/YYYY'),'F','VIA
GENOVA','22','80200','CASERTA');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111116','ALESSIO','SOMMA',TO_DATE('30/08/1983','DD/MM/YYYY'),'M','VIA
MANDARINO','82','80200','CASERTA');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111117','DANIELE','SICILIANO',TO_DATE('01/02/1976','DD/MM/YYYY'),'M','VIA
TASSO','66','80400','BENEVENTO');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111118','GENNARO','ESPOSITO',TO_DATE('19/01/1977','DD/MM/YYYY'),'M','CORSO
GARIBALDI','452','80100','NAPOLI');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111119','MANILA','LANZI',TO_DATE('31/12/1968','DD/MM/YYYY'),'F','VIA
COLOMBO','6','80400','BENEVENTO');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111120','SIMONE','LAURENTI',TO_DATE('21/11/1982','DD/MM/YYYY'),'M','VIA
DRAGHI','72','80300','AVELLINO');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111121','ALFREDO','BENGALONI',TO_DATE('11/08/1982','DD/MM/YYYY'),'M','VIA
SODOMA','32','80500','SALERNO');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111122','FRANCESCO','AMICO',TO_DATE('28/04/1975','DD/MM/YYYY'),'M','VIA
BARI','15','80100','NAPOLI');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111123','CHIARA','NASONE',TO_DATE('14/01/1970','DD/MM/YYYY'),'F','VIA
BANFI','72','80100','NAPOLI');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111124','ANDREA','GALLO',TO_DATE('06/06/1978','DD/MM/YYYY'),'M','VIA
NOBILE','72','80500','SALERNO');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111125','CAMILLO','BENSO',TO_DATE('17/05/1965','DD/MM/YYYY'),'M','VIA
CAVOUR','49','80200','CASERTA');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111126','ANGELA','CASCIO',TO_DATE('8/10/1988','DD/MM/YYYY'),'F','VIALE DELLE
FONTANELLE','78','80300','AVELLINO');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111127','GIOVANNI','CANGIANO',TO_DATE('19/07/1973','DD/MM/YYYY'),'M','VIALE DEI
FIORI','18','80400','BENEVENTO');
```

```
INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111128','LUCA','ABBRUZZESE',TO_DATE('14/08/1980','DD/MM/YYYY'),'M','VIA
BIFULCO','62','80400','BENEVENTO');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111129','GIANNI','CAIARULO',TO_DATE('15/02/1978','DD/MM/YYYY'),'M','VIA  DEL
CONTE','78','80100','NAPOLI');

INSERT INTO DOCENTE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA) VALUES
('111130','GIORGIO','LOMBARDO',TO_DATE('30/04/1968','DD/MM/YYYY'),'M','VIA DEI
SORDI','302','80300','AVELLINO');
```

Studente

```
INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111150','MARIO','CICALONE',TO_DATE('19/08/1999','DD/MM/YYYY'),'M','VIA
CANTELMO','29','80100','NAPOLI', '0124', TO_DATE('01/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111151','MARIA','BIANCHI',TO_DATE('30/05/1998','DD/MM/YYYY'),'F','VIA
CASILLO','33','80100','NAPOLI', '0332', TO_DATE('01/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111152','ANDREA','ROSSI',TO_DATE('19/08/1999','DD/MM/YYYY'),'M','VIA
COLOMBO','32','80200','CASERTA', '0328', TO_DATE('01/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111153','ANNA','BRUSAFERRO',TO_DATE('15/05/1997','DD/MM/YYYY'),'F','VIALE DEGLI
OLEANDRI','21','80500','SALERNO', '0327', TO_DATE('01/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111154','LUCA','LONGHI',TO_DATE('30/01/2000','DD/MM/YYYY'),'M','VIA
MARINAIO','63','80100','NAPOLI', '0328', TO_DATE('15/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111155','LAURA','FOSSETTA',TO_DATE('19/08/1999','DD/MM/YYYY'),'F','VIA
VENEZIA','16','80400','BENEVENTO', '0512', TO_DATE('06/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111156','MATTEO','TROMBA',TO_DATE('09/07/1998','DD/MM/YYYY'),'M','VIA
SICARI','29','80300','AVELLINO', '0122', TO_DATE('07/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111157','MARIO','STROPPA',TO_DATE('19/08/1999','DD/MM/YYYY'),'M','VIA
GOGNA','99','80100','NAPOLI', '0126', TO_DATE('18/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111159','GIUSEPPE','CONTI',TO_DATE('12/11/1996','DD/MM/YYYY'),'M','VIA
GIUSTIZIA','39','80400','BENEVENTO', '0120', TO_DATE('01/09/2019','DD/MM/YYYY')) ;

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111160','CAMILLA','NOLA',TO_DATE('22/03/1996','DD/MM/YYYY'),'F','VIA
SONDRIO','11','80200','CASERTA', '0331', TO_DATE('01/08/2019','DD/MM/YYYY'));
```

```
INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111161','MARIA','BROCCOLI',TO_DATE('17/08/1998','DD/MM/YYYY'),'F','VIA
ORAZIO','29','80100','NAPOLI', '0330', TO_DATE('02/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111162','VINCENZO','MOTTA',TO_DATE('17/02/1996','DD/MM/YYYY'),'M','VIA
GARIBALDI','66','80100','NAPOLI', '0326', TO_DATE('22/07/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111163','CARLO','MASCARA',TO_DATE('09/10/1993','DD/MM/YYYY'),'M','VIA
SPLENDENTE','32','80500','SALERNO', '0514', TO_DATE('16/07/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111164','LORENZO','MANZO',TO_DATE('19/08/1999','DD/MM/YYYY'),'M','VIA
TORINO','22','80100','NAPOLI', '0124', TO_DATE('06/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111165','ANNA','MANITA',TO_DATE('15/05/1994','DD/MM/YYYY'),'F','VIA
LEVI','31','80500','SALERNO', '0125', TO_DATE('02/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111166','CARMINE','SALE',TO_DATE('17/05/1998','DD/MM/YYYY'),'M','VIALE DEGLI
OLEANDRI','21','80100','NAPOLI', '0123', TO_DATE('01/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111167','LORENZO','MUSETTI',TO_DATE('23/08/1998','DD/MM/YYYY'),'M','VIA
NAPOLI','22','80500','SALERNO', '0121', TO_DATE('06/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111168','ANTONIO','CARONTE',TO_DATE('22/03/1996','DD/MM/YYYY'),'M','VIA
GENOVA','77','80200','CASERTA', '0328', TO_DATE('01/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111169','GIUSEPPE','ANDREOLLI',TO_DATE('15/07/1995','DD/MM/YYYY'),'M','VIA
BARI','49','80400','BENEVENTO', '0326', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111170','MARCO','BOCCIA',TO_DATE('10/02/1994','DD/MM/YYYY'),'M','VIA DEGLI
OREFICI','99','80100','NAPOLI', '0330', TO_DATE('02/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111171','LORENZO','SCAMPI',TO_DATE('19/08/1999','DD/MM/YYYY'),'M','VIA
TORINO','22','80200','CASERTA', '0512', TO_DATE('06/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111172','GIANLUCA','SORDI',TO_DATE('04/02/1999','DD/MM/YYYY'),'M','VIA DEL
SANTO','62','80100','NAPOLI', '0515', TO_DATE('06/08/2019','DD/MM/YYYY'));

INSERT INTO STUDENTE(MATRICOLA_STUDENTE, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, CODICE_CORSO, DATA_ISCRIZIONE) VALUES
('111173','AJEJE','BRAZORF',TO_DATE('22/04/1996','DD/MM/YYYY'),'M','VIA
TARANTO','36','80500','SALERNO', '0515', TO_DATE('06/08/2019','DD/MM/YYYY'));
```

Insegna Edizione

```
INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('SNN', TO_DATE('2019', 'YYYY'), '111111', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('ASD', TO_DATE('2019', 'YYYY'), '111112', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('BD', TO_DATE('2019', 'YYYY'), '111113', 'TEORIA E LABORATORIO');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('BIOC', TO_DATE('2019', 'YYYY'), '111114', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('ASD', TO_DATE('2019', 'YYYY'), '111113', 'LABORATORIO');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('BIOC', TO_DATE('2019', 'YYYY'), '111115', 'LABORATORIO');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('CHIMG', TO_DATE('2019', 'YYYY'), '111115', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('CHIMG', TO_DATE('2019', 'YYYY'), '111116', 'LABORATORIO');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('OCEA', TO_DATE('2019', 'YYYY'), '111117', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('METEO', TO_DATE('2019', 'YYYY'), '111118', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('MACL', TO_DATE('2019', 'YYYY'), '111112', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('COMG', TO_DATE('2019', 'YYYY'), '111119', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('BIODIV', TO_DATE('2019', 'YYYY'), '111121', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('METEOA', TO_DATE('2019', 'YYYY'), '111118', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('NAVSAT', TO_DATE('2019', 'YYYY'), '111122', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('MICROB', TO_DATE('2019', 'YYYY'), '111120', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('SCIES', TO_DATE('2019', 'YYYY'), '111123', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('IMPE', TO_DATE('2019', 'YYYY'), '111124', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('MACC', TO_DATE('2019', 'YYYY'), '111125', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('IDRA', TO_DATE('2019', 'YYYY'), '111126', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('CIDRA', TO_DATE('2019', 'YYYY'), '111126', 'TEORIA');
```

```
INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('PROGCE', TO_DATE('2019', 'YYYY'), '111119', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('ELETT', TO_DATE('2019', 'YYYY'), '111127', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('OCEAC', TO_DATE('2019', 'YYYY'), '111117', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('ENED', TO_DATE('2019', 'YYYY'), '111124', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('RTI', TO_DATE('2019', 'YYYY'), '111128', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('TI', TO_DATE('2019', 'YYYY'), '111128', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('GSE', TO_DATE('2019', 'YYYY'), '111124', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('AI', TO_DATE('2019', 'YYYY'), '111127', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('END', TO_DATE('2019', 'YYYY'), '111129', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('NEU', TO_DATE('2019', 'YYYY'), '111121', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('AEA', TO_DATE('2019', 'YYYY'), '111129', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('BPAS', TO_DATE('2019', 'YYYY'), '111130', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('FISGE', TO_DATE('2019', 'YYYY'), '111130', 'TEORIA');

INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, TESSERINO_DOCENTE,
TIPO_DOCENTE) VALUES ('MMSD', TO_DATE('2019', 'YYYY'), '111120', 'TEORIA');
```

Frequenta Edizione Insegnamento

```
INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'SNN', '111165', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'ASD', '111150', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'BD', '111150', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'BIOC', '111166', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'CHIMG', '111166', TO_DATE('01/09/2019','DD/MM/YYYY'));
```

```
INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'OCEA', '111156', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'METEO', '111156', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'MACL', '111159', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'COMG', '111159', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'BIODIV', '111157', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'METEOA', '111167', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'NAVSAT', '111167', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'MICROB', '111157', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'SCIES', '111157', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'IMPE', '111154', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'MACC', '111168', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'IDRA', '111151', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'CIDRA', '111151', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'PROGCE', '111162', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'ELETT', '111169', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'OCEAC', '111160', TO_DATE('01/09/2019','DD/MM/YYYY'));
```

```
INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'ENED', '111160', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'RTI', '111161', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'TI', '111170', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'GSE', '111151', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'AI', '111151', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'END', '111155', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'NEU', '111171', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'AEA', '111172', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'BPAS', '111172', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'MACC', '111152', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'IMPE', '111152', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'FISGE', '111163', TO_DATE('01/09/2019','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'MMSD', '111163', TO_DATE('01/03/2020','DD/MM/YYYY'));

INSERT INTO FREQUENTA_EDIZIONE_INSEGNAMENTO(ANNO_ACCADEMICO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, DATA_INS) VALUES
(TO_DATE('2019', 'YYYY'), 'METEO', '111165', TO_DATE('01/03/2020','DD/MM/YYYY'));
```

Azienda

```
INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000002', 'MANPOWER', 'VIA TIRABUCCHI', '65', '80300', 'AVELLINO');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000003', 'BEANTECH SRL', 'VIA CAMILLERI', '65', '80400', 'BENEVENTO');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000004', 'FASHION JOBS', 'VIA SCATURCHIO', '65', '80500', 'SALERNO');
```

```
INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000005', '4DAYS SRL', 'VIA TRIESTE', '96', '80100', 'NAPOLI');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000006', 'LAMORETTI', 'VIA ASTI', '55', '80200', 'CASERTA');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000007', 'JUROP SPA', 'VIA DEL CORONEO', '19', '80500', 'SALERNO');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000008', 'JOB INFORMATICA', 'VIA SACCHI', '55', '80100', 'NAPOLI');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000009', 'LUMIT SPA', 'VIA TASSO', '60', '80100', 'NAPOLI');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000010', 'MASERATI SPA', 'VIA CAVALLI', '51', '80300', 'AVELLINO');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000011', 'MIPOT SPA', 'VIA FILANGIERI', '35', '80400', 'BENEVENTO');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000012', 'MONOGRID SRL', 'VIA SALLUSTI', '98', '80200', 'CASERTA');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000013', 'MULTITEMA SRL', 'VIA MANZONI', '45', '80400', 'BENEVENTO');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000014', 'PALMA SPA', 'VIA SALOMONE', '56', '80300', 'AVELLINO');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000015', 'SDS SRL', 'VIA BERNA', '10', '80500', 'SALERNO');

INSERT INTO AZIENDA(PARTITA_IVA, NOME, VIA, NUMERO_CIVICO, CAP, CITTA) VALUES
('800000000000001', 'RANDSTAD', 'VIA BERNINI', '55', '80200', 'CASERTA');
```

Tutor Aziendale

```
INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111201','LUCA','BOLLE',TO_DATE('16/07/1972','DD/MM/YYYY'),'M','VIA DE
MILLE','62','80100','NAPOLI', '800000000000002');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111202','MATTEO','BORGHI',TO_DATE('24/04/1981','DD/MM/YYYY'),'M','VIA
SASSI','54','80300','AVELLINO', '800000000000002');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111203','MASSIMO','GALLIOTTI',TO_DATE('03/03/1973','DD/MM/YYYY'),'M','VIA
BARI','32','80200','CASERTA', '800000000000003');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111204','DANIELE','CURTI',TO_DATE('19/02/1977','DD/MM/YYYY'),'M','VIA
CAVOUR','151','80400','BENEVENTO', '800000000000004');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111205','ANGELO','POSA',TO_DATE('11/09/1975','DD/MM/YYYY'),'M','VIA
SASTRI','12','80500','SALERNO', '800000000000005');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
```

```
('111206','ANDREA','MARINI',TO_DATE('08/06/1970','DD/MM/YYYY'),'F','VIA DEI
CARDINALI','03','80300','AVELLINO', '800000000000006');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111207','FRANCO','MONTE',TO_DATE('10/10/1976','DD/MM/YYYY'),'M','VIA
SOLIMENA','99','80200','CASERTA', '800000000000007');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111208','SANTO','BOSCO',TO_DATE('18/06/1969','DD/MM/YYYY'),'M','VIA DE
ROVERI','115','80100','NAPOLI', '800000000000008');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111209','DARIO','MONTI',TO_DATE('29/05/1976','DD/MM/YYYY'),'M','VIA
MOCCIA','125','80100','NAPOLI', '800000000000009');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111210','MARIALUISA','BOSCO',TO_DATE('19/09/1979','DD/MM/YYYY'),'F','VIA
BALAUSTRA','11','80100','NAPOLI', '800000000000010');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111211','GIANNI','CODA',TO_DATE('07/03/1972','DD/MM/YYYY'),'M','VIA
FIRENZE','22','80200','CASERTA', '800000000000011');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111212','SALVATORE','TROTTA',TO_DATE('28/04/1974','DD/MM/YYYY'),'M','VIA
PADOVA','67','80100','NAPOLI', '800000000000012');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111213','GIUSY','MARINO',TO_DATE('02/06/1979','DD/MM/YYYY'),'F','VIA DELLA
FORTUNA','50','80400','BENEVENTO', '800000000000013');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111214','GIOVANNA','SOTTILE',TO_DATE('28/12/1972','DD/MM/YYYY'),'F','VIA
SALOMONE','15','80300','AVELLINO', '800000000000014');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111215','SANTA','FALCHI',TO_DATE('11/01/1971','DD/MM/YYYY'),'F','VIA
SCAVULLI','58','80500','SALERNO', '800000000000015');

INSERT INTO TUTOR_AZIENDALE(NUMERO_TESSERINO, NOME, COGNOME, DATA_NASCITA, SESSO, VIA,
NUMERO_CIVICO, CAP, CITTA, PARTITA_IVA) VALUES
('111200','ANDREA','GALLIOTTI',TO_DATE('08/06/1970','DD/MM/YYYY'),'M','VIA DE
GASPARI','62','80500','SALERNO', '800000000000001');
```

Tirocinio

```
INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111311', 12, TO_DATE('01/01/2020','DD/MM/YYYY'),
TO_DATE('30/05/2020','DD/MM/YYYY'), '111126', '111212', '111150');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111312', 9, TO_DATE('01/01/2020','DD/MM/YYYY'),
TO_DATE('30/03/2020','DD/MM/YYYY'), '111128', '111208', '111151');
```

```
INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111313', 12, TO_DATE('01/03/2020','DD/MM/YYYY'),
TO_DATE('30/05/2020','DD/MM/YYYY'), '111127', '111209', '111152');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111314', 12, TO_DATE('01/01/2020','DD/MM/YYYY'),
TO_DATE('30/08/2020','DD/MM/YYYY'), '111125', '111210', '111153');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111315', 9, TO_DATE('01/04/2020','DD/MM/YYYY'),
TO_DATE('30/07/2020','DD/MM/YYYY'), '111124', '111211', '111154');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111316', 12, TO_DATE('01/01/2020','DD/MM/YYYY'),
TO_DATE('30/06/2020','DD/MM/YYYY'), '111124', '111213', '111155');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111317',6, TO_DATE('01/01/2020','DD/MM/YYYY'),
TO_DATE('30/03/2020','DD/MM/YYYY'), '111123', '111214', '111156');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111318', 9, TO_DATE('01/06/2020','DD/MM/YYYY'),
TO_DATE('30/09/2020','DD/MM/YYYY'), '111122', '111215', '111157');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111319', 12, TO_DATE('01/01/2020','DD/MM/YYYY'),
TO_DATE('30/06/2020','DD/MM/YYYY'), '111121', '111207', '111165');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111320', 12, TO_DATE('01/04/2020','DD/MM/YYYY'),
TO_DATE('30/07/2020','DD/MM/YYYY'), '111120', '111212', '111159');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111321', 6, TO_DATE('01/03/2020','DD/MM/YYYY'),
TO_DATE('30/08/2020','DD/MM/YYYY'), '111119', '111206', '111160');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111322', 6, TO_DATE('01/09/2020','DD/MM/YYYY'),
TO_DATE('30/11/2020','DD/MM/YYYY'), '111119', '111205', '111166');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111323', 12, TO_DATE('01/02/2020','DD/MM/YYYY'),
TO_DATE('30/05/2020','DD/MM/YYYY'), '111118', '111204', '111161');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111324', 9, TO_DATE('01/01/2020','DD/MM/YYYY'),
TO_DATE('30/03/2020','DD/MM/YYYY'), '111117', '111203', '111162');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111325', 12, TO_DATE('01/01/2020','DD/MM/YYYY'),
TO_DATE('30/06/2020','DD/MM/YYYY'), '111117', '111202', '111163');
```

```
INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111326', 6, TO_DATE('01/07/2020','DD/MM/YYYY'),
TO_DATE('30/08/2020','DD/MM/YYYY'), '111116', '111201', '111164');

INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
('111111111111327', 6, TO_DATE('01/09/2020','DD/MM/YYYY'),
TO_DATE('30/11/2020','DD/MM/YYYY'), '111115', '111201', '111167');
```

Questionario

```
INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111111', 'SNN', TO_DATE('2019','YYYY'), 4, 5, 3, 2, '111111', '111165',
TO_DATE('30/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111112', 'ASD', TO_DATE('2019','YYYY'), 4, 4, 4, 4, '111112', '111150',
TO_DATE('20/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111113', 'BD', TO_DATE('2019','YYYY'), 4, 2, 3, 2, '111113', '111150',
TO_DATE('28/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111114', 'BIOC', TO_DATE('2019','YYYY'), 5, 5, 2, 5, '111114', '111166',
TO_DATE('26/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111115', 'ASD', TO_DATE('2019','YYYY'), 1, 5, 5, 2, '111113', '111150',
TO_DATE('22/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111116', 'CHIMG', TO_DATE('2019','YYYY'), 4, 5, 3, 3, '111115', '111166',
TO_DATE('25/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111117', 'CHIMG', TO_DATE('2019','YYYY'), 3, 3, 3, 4, '111116', '111166',
TO_DATE('26/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111118', 'OCEA', TO_DATE('2019','YYYY'), 4, 2, 2, 3, '111117', '111156',
TO_DATE('22/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111119', 'METEO', TO_DATE('2019','YYYY'), 1, 2, 1, 2, '111118', '111156',
TO_DATE('28/05/2020','DD/MM/YYYY'));
```

```
INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111120', 'MACL', TO_DATE('2019','YYYY'), 3, 4, 4, 4, '111119', '111159',
TO_DATE('25/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111121', 'COMG', TO_DATE('2019','YYYY'), 2, 2, 4, 5, '111111', '111159',
TO_DATE('30/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111122', 'BIODIV', TO_DATE('2019','YYYY'), 3, 4, 3, 2, '111121', '111157',
TO_DATE('30/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111123', 'METEOA', TO_DATE('2019','YYYY'), 5, 5, 3, 2, '111118', '111167',
TO_DATE('19/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111124', 'NAVSAT', TO_DATE('2019','YYYY'), 4, 5, 1, 2, '111122', '111167',
TO_DATE('11/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111125', 'MICROB', TO_DATE('2019','YYYY'), 2, 2, 3, 2, '111120', '111157',
TO_DATE('24/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111126', 'SCIES', TO_DATE('2019','YYYY'), 4, 2, 1, 2, '111123', '111157',
TO_DATE('26/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111127', 'IMPE', TO_DATE('2019','YYYY'), 1, 1, 3, 2, '111124', '111154',
TO_DATE('27/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111128', 'MACC', TO_DATE('2019','YYYY'), 4, 3, 1, 2, '111125', '111152',
TO_DATE('25/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111129', 'IDRA', TO_DATE('2019','YYYY'), 2, 2, 3, 2, '111126', '111151',
TO_DATE('26/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111130', 'CIDRA', TO_DATE('2019','YYYY'), 1, 2, 3, 2, '111125', '111151',
TO_DATE('19/05/2020','DD/MM/YYYY'));
```

```
INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111131', 'PROGCE', TO_DATE('2019','YYYY'), 4, 5, 4, 4, '111119', '111162',
TO_DATE('24/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111132', 'ELETT', TO_DATE('2019','YYYY'), 4, 5, 5, 5, '111127', '111162',
TO_DATE('22/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111133', 'OCEAC', TO_DATE('2019','YYYY'), 1, 1, 1, 2, '111117', '111160',
TO_DATE('21/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111134', 'ENED', TO_DATE('2019','YYYY'), 3, 3, 3, 2, '111124', '111160',
TO_DATE('26/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111135', 'RTI', TO_DATE('2019','YYYY'), 4, 4, 3, 2, '111128', '111161',
TO_DATE('27/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111136', 'TI', TO_DATE('2019','YYYY'), 4, 1, 1, 2, '111124', '111161',
TO_DATE('27/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111137', 'GSE', TO_DATE('2019','YYYY'), 2, 2, 2, 2, '111127', '111151',
TO_DATE('26/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111138', 'AI', TO_DATE('2019','YYYY'), 3, 3, 3, 3, '111129', '111151',
TO_DATE('20/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111139', 'END', TO_DATE('2019','YYYY'), 4, 4, 3, 2, '111121', '111155',
TO_DATE('18/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111140', 'NEU', TO_DATE('2019','YYYY'), 4, 2, 3, 2, '111129', '111155',
TO_DATE('16/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111144', 'AEA', TO_DATE('2019','YYYY'), 4, 2, 2, 2, '111129', '111172',
TO_DATE('28/05/2020','DD/MM/YYYY'));
```

```
INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111141', 'BPAS', TO_DATE('2019','YYYY'), 4, 3, 3, 5, '111130', '111172',
TO_DATE('23/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111142', 'FISGE', TO_DATE('2019','YYYY'), 4, 5, 4, 5, '111130', '111163',
TO_DATE('21/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111143', 'MMSD', TO_DATE('2019','YYYY'), 4, 4, 3, 5, '111120', '111163',
TO_DATE('18/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111144', 'METEO', TO_DATE('2019','YYYY'), 1, 2, 1, 2, '111118', '111165',
TO_DATE('28/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111145', 'ASD', TO_DATE('2019','YYYY'), 4, 5, 4, 5, '111113', '111153',
TO_DATE('21/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111146', 'BIOC', TO_DATE('2019','YYYY'), 4, 4, 3, 5, '111115', '111153',
TO_DATE('18/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111147', 'IDRA', TO_DATE('2019','YYYY'), 2, 2, 3, 2, '111125', '111152',
TO_DATE('26/05/2020','DD/MM/YYYY'));

INSERT INTO QUESTIONARIO(NUMERO_QUESTIONARIO, CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
MATERIALE_DIDATTICO, GRADIMENTO, DISPONIBILITA_DOCENTE, PRECISIONE_ORARIO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_COMPILAZIONE) VALUES
('111148', 'CIDRA', TO_DATE('2019','YYYY'), 1, 2, 3, 2, '111126', '111152',
TO_DATE('19/05/2020','DD/MM/YYYY'));
```

Appello

```
INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'SNN',
TO_DATE('11/06/2020','DD/MM/YYYY'),TO_DATE('24/06/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/06/2020','DD/MM/YYYY'), 'ASD',
TO_DATE('01/06/2020','DD/MM/YYYY'),TO_DATE('14/06/2020','DD/MM/YYYY'), 20,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('23/07/2020','DD/MM/YYYY'), 'BD',
TO_DATE('21/07/2020','DD/MM/YYYY'),TO_DATE('22/07/2020','DD/MM/YYYY'), 15,'SCRITTO');
```

```
INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('17/06/2020','DD/MM/YYYY'), 'BIOC',
TO_DATE('01/06/2020','DD/MM/YYYY'),TO_DATE('16/06/2020','DD/MM/YYYY'), 15,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/06/2020','DD/MM/YYYY'), 'CHIMG',
TO_DATE('02/06/2020','DD/MM/YYYY'),TO_DATE('14/06/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('28/07/2020','DD/MM/YYYY'), 'METEO',
TO_DATE('14/07/2020','DD/MM/YYYY'),TO_DATE('27/07/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('29/07/2020','DD/MM/YYYY'), 'METEO',
TO_DATE('14/07/2020','DD/MM/YYYY'),TO_DATE('27/07/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('23/06/2020','DD/MM/YYYY'), 'OCEA',
TO_DATE('01/06/2020','DD/MM/YYYY'),TO_DATE('20/06/2020','DD/MM/YYYY'), 10,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/07/2020','DD/MM/YYYY'), 'MACL',
TO_DATE('01/07/2020','DD/MM/YYYY'),TO_DATE('14/07/2020','DD/MM/YYYY'), 10,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('29/06/2020','DD/MM/YYYY'), 'COMG',
TO_DATE('15/06/2020','DD/MM/YYYY'),TO_DATE('28/06/2020','DD/MM/YYYY'), 20,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/06/2020','DD/MM/YYYY'), 'BIODIV',
TO_DATE('01/06/2020','DD/MM/YYYY'),TO_DATE('20/06/2020','DD/MM/YYYY'), 10,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('24/06/2020','DD/MM/YYYY'), 'METEOA',
TO_DATE('10/06/2020','DD/MM/YYYY'),TO_DATE('20/06/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('29/07/2020','DD/MM/YYYY'), 'NAVSAT',
TO_DATE('10/07/2020','DD/MM/YYYY'),TO_DATE('28/07/2020','DD/MM/YYYY'), 20,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/06/2020','DD/MM/YYYY'), 'MICROB',
TO_DATE('01/06/2020','DD/MM/YYYY'),TO_DATE('21/06/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('04/07/2020','DD/MM/YYYY'), 'SCIES',
TO_DATE('20/06/2020','DD/MM/YYYY'),TO_DATE('03/07/2020','DD/MM/YYYY'), 15,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('01/07/2020','DD/MM/YYYY'), 'IMPE',
TO_DATE('20/06/2020','DD/MM/YYYY'),TO_DATE('30/06/2020','DD/MM/YYYY'), 20,'SCRITTO');
```

```
INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('27/06/2020','DD/MM/YYYY'), 'MACC',
TO_DATE('15/06/2020','DD/MM/YYYY'),TO_DATE('26/06/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'IDRA',
TO_DATE('15/06/2020','DD/MM/YYYY'),TO_DATE('29/06/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('11/07/2020','DD/MM/YYYY'), 'CIDRA',
TO_DATE('28/06/2020','DD/MM/YYYY'),TO_DATE('10/07/2020','DD/MM/YYYY'), 20,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/07/2020','DD/MM/YYYY'), 'PROGCE',
TO_DATE('01/07/2020','DD/MM/YYYY'),TO_DATE('21/07/2020','DD/MM/YYYY'), 15,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('17/06/2020','DD/MM/YYYY'), 'ELETT',
TO_DATE('01/06/2020','DD/MM/YYYY'),TO_DATE('16/06/2020','DD/MM/YYYY'), 20,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('23/07/2020','DD/MM/YYYY'), 'OCEAC',
TO_DATE('13/07/2020','DD/MM/YYYY'),TO_DATE('22/07/2020','DD/MM/YYYY'), 15,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('19/06/2020','DD/MM/YYYY'), 'ENED',
TO_DATE('01/06/2020','DD/MM/YYYY'),TO_DATE('18/06/2020','DD/MM/YYYY'), 20,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('10/07/2020','DD/MM/YYYY'), 'RTI',
TO_DATE('01/07/2020','DD/MM/YYYY'),TO_DATE('09/07/2020','DD/MM/YYYY'), 10,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('14/06/2020','DD/MM/YYYY'), 'TI',
TO_DATE('01/06/2020','DD/MM/YYYY'),TO_DATE('13/06/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('25/06/2020','DD/MM/YYYY'), 'GSE',
TO_DATE('15/06/2020','DD/MM/YYYY'),TO_DATE('24/06/2020','DD/MM/YYYY'), 15,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('18/07/2020','DD/MM/YYYY'), 'AI',
TO_DATE('01/07/2020','DD/MM/YYYY'),TO_DATE('17/07/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'END',
TO_DATE('11/06/2020','DD/MM/YYYY'),TO_DATE('27/06/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/07/2020','DD/MM/YYYY'), 'NEU',
TO_DATE('15/06/2020','DD/MM/YYYY'),TO_DATE('28/07/2020','DD/MM/YYYY'), 20,'ORALE');
```

```
INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('28/06/2020','DD/MM/YYYY'), 'AEA',
TO_DATE('14/06/2020','DD/MM/YYYY'),TO_DATE('27/06/2020','DD/MM/YYYY'), 10,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('28/07/2020','DD/MM/YYYY'), 'BPAS',
TO_DATE('15/06/2020','DD/MM/YYYY'),TO_DATE('27/07/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'FISGE',
TO_DATE('15/06/2020','DD/MM/YYYY'),TO_DATE('27/06/2020','DD/MM/YYYY'), 10,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/07/2020','DD/MM/YYYY'), 'MMSD',
TO_DATE('01/07/2020','DD/MM/YYYY'),TO_DATE('14/07/2020','DD/MM/YYYY'), 20,'ORALE');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'ASD',
TO_DATE('15/06/2020','DD/MM/YYYY'),TO_DATE('27/06/2020','DD/MM/YYYY'), 10,'SCRITTO');

INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO, DATA_INIZIO,
DATA_FINE, MAX_STUDENTI, TIPO) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/07/2020','DD/MM/YYYY'), 'BIOC',
TO_DATE('01/07/2020','DD/MM/YYYY'),TO_DATE('14/07/2020','DD/MM/YYYY'), 20,'ORALE');
```

Presiede Appello

```
INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'SNN', '111111', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/06/2020','DD/MM/YYYY'), 'ASD', '111112', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('23/07/2020','DD/MM/YYYY'), 'BD', '111113', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('17/06/2020','DD/MM/YYYY'), 'BIOC', '111114', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'ASD', '111113', 'N');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('17/06/2020','DD/MM/YYYY'), 'BIOC', '111115', 'N');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/06/2020','DD/MM/YYYY'), 'CHIMG', '111115', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/06/2020','DD/MM/YYYY'), 'CHIMG', '111116', 'N');
```

```
INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('23/06/2020','DD/MM/YYYY'), 'OCEA', '111117', 'N');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('28/07/2020','DD/MM/YYYY'), 'METEO', '111118', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('29/07/2020','DD/MM/YYYY'), 'METEO', '111118', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/07/2020','DD/MM/YYYY'), 'MACL', '111112', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('29/06/2020','DD/MM/YYYY'), 'COMG', '111119', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/06/2020','DD/MM/YYYY'), 'BIODIV', '111121', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('24/06/2020','DD/MM/YYYY'), 'METEOA', '111118', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('29/07/2020','DD/MM/YYYY'), 'NAVSAT', '111122', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/06/2020','DD/MM/YYYY'), 'MICROB', '111120', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('04/07/2020','DD/MM/YYYY'), 'SCIES', '111123', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('01/07/2020','DD/MM/YYYY'), 'IMPE', '111124', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('27/06/2020','DD/MM/YYYY'), 'MACC', '111125', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'IDRA', '111125', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('11/07/2020','DD/MM/YYYY'), 'CIDRA', '111126', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/07/2020','DD/MM/YYYY'), 'PROGCE', '111119', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('17/06/2020','DD/MM/YYYY'), 'ELETT', '111127', 'Y');
```

```
INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('23/07/2020','DD/MM/YYYY'), 'OCEAC', '111117', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('19/06/2020','DD/MM/YYYY'), 'ENED', '111124', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('10/07/2020','DD/MM/YYYY'), 'RTI', '111128', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('14/06/2020','DD/MM/YYYY'), 'TI', '111128', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('25/06/2020','DD/MM/YYYY'), 'GSE', '111124', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('18/07/2020','DD/MM/YYYY'), 'AI', '111127', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'END', '111129', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/07/2020','DD/MM/YYYY'), 'NEU', '111121', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('28/06/2020','DD/MM/YYYY'), 'AEA', '111129', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('28/07/2020','DD/MM/YYYY'), 'BPAS', '111130', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'FISGE', '111130', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/07/2020','DD/MM/YYYY'), 'MMSD', '111120', 'Y');

INSERT INTO PRESIEDE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
TESSERINO_DOCENTE, PRESIDENTE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/07/2020','DD/MM/YYYY'), 'BIOC', '111115', 'Y');
```

Prenotazione Appello

```
INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'SNN', '111165', '1111',
TO_DATE('23/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/06/2020','DD/MM/YYYY'), 'ASD', '111150', '1111',
TO_DATE('11/06/2020','DD/MM/YYYY'));
```

```
INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('23/07/2020','DD/MM/YYYY'), 'BD', '111150', '1111',
TO_DATE('11/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('17/06/2020','DD/MM/YYYY'), 'BIOC', '111166', '1111',
TO_DATE('09/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/06/2020','DD/MM/YYYY'), 'CHIMG', '111166', '1111',
TO_DATE('09/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('23/06/2020','DD/MM/YYYY'), 'OCEA', '111156', '1111',
TO_DATE('17/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('28/07/2020','DD/MM/YYYY'), 'METEO', '111156', '1111',
TO_DATE('20/07/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/07/2020','DD/MM/YYYY'), 'MACL', '111159', '1111',
TO_DATE('15/07/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('29/06/2020','DD/MM/YYYY'), 'COMG', '111159', '1111',
TO_DATE('20/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/06/2020','DD/MM/YYYY'), 'BIODIV', '111157', '1111',
TO_DATE('14/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('24/06/2020','DD/MM/YYYY'), 'METEOA', '111167', '1111',
TO_DATE('16/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('29/07/2020','DD/MM/YYYY'), 'NAVSAT', '111167', '1111',
TO_DATE('21/07/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('01/07/2020','DD/MM/YYYY'), 'IMPE', '111152', '1111',
TO_DATE('26/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('27/06/2020','DD/MM/YYYY'), 'MACC', '111152', '1111',
TO_DATE('18/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'IDRA', '111151', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));
```

```
INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('11/07/2020','DD/MM/YYYY'), 'CIDRA', '111151', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('22/07/2020','DD/MM/YYYY'), 'PROGCE', '111162', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('17/06/2020','DD/MM/YYYY'), 'ELETT', '111162', '1111',
TO_DATE('14/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('10/07/2020','DD/MM/YYYY'), 'RTI', '111161', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('14/06/2020','DD/MM/YYYY'), 'TI', '111161', '1111',
TO_DATE('04/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('23/07/2020','DD/MM/YYYY'), 'OCEAC', '111160', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('19/06/2020','DD/MM/YYYY'), 'ENED', '111160', '1111',
TO_DATE('10/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('28/06/2020','DD/MM/YYYY'), 'AEA', '111172', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('28/07/2020','DD/MM/YYYY'), 'BPAS', '111172', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'FISGE', '111163', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/07/2020','DD/MM/YYYY'), 'MMSD', '111163', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('29/07/2020','DD/MM/YYYY'), 'METEO', '111165', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('25/06/2020','DD/MM/YYYY'), 'GSE', '111151', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));
```

```
INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('18/07/2020','DD/MM/YYYY'), 'AI', '111151', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'ASD', '111153', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('15/07/2020','DD/MM/YYYY'), 'BIOC', '111153', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/06/2020','DD/MM/YYYY'), 'END', '111155', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));

INSERT INTO PRENOTAZIONE_APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
MATRICOLA_STUDENTE, NUMERO_PRENOTAZIONE, DATA_PRENOTAZIONE) VALUES
(TO_DATE('2019','YYYY'), TO_DATE('30/07/2020','DD/MM/YYYY'), 'NEU', '111155', '1111',
TO_DATE('24/06/2020','DD/MM/YYYY'));
```

Esame Superato

```
INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311111', 27, 'N', TO_DATE('30/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'SNN', '111165');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311112', 23, 'N', TO_DATE('15/06/2020','DD/MM/YYYY'),TO_DATE('2019','YYYY'),
'ASD', '111150');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311113', 25, 'N', TO_DATE('23/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'BD', '111150');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311114', 28, 'N',TO_DATE('17/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'BIOC', '111166');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311116', 30, 'N',TO_DATE('15/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'CHIMG', '111166');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311117', 27, 'N',TO_DATE('23/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'OCEA', '111156');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311118', 26, 'N', TO_DATE('28/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'METEO', '111156');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311119', 30, 'Y', TO_DATE('22/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'MACL', '111159');
```

```
INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311120', 30, 'Y', TO_DATE('29/06/2020','DD/MM/YYYY'),TO_DATE('2019','YYYY'),
'COMG', '111159');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311121', 27, 'N', TO_DATE('22/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'BIODIV', '111157');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311122', 22, 'N', TO_DATE('24/06/2020','DD/MM/YYYY'),TO_DATE('2019','YYYY'),
'METEOA', '111167');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311123', 21, 'N',TO_DATE('29/07/2020','DD/MM/YYYY'),TO_DATE('2019','YYYY'),
'NAVSAT', '111167');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311124', 26, 'N',TO_DATE('01/07/2020','DD/MM/YYYY'),TO_DATE('2019','YYYY'),
'IMPE', '111152');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311125', 28, 'N',TO_DATE('27/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'MACC', '111152');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311126', 28, 'N',TO_DATE('30/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'IDRA', '111151');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311127', 28, 'N',TO_DATE('11/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'CIDRA', '111151');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311128', 22, 'N',TO_DATE('22/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'PROGCE', '111162');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311129', 26, 'N',TO_DATE('17/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'ELETT', '111162');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311130', 25, 'N',TO_DATE('10/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'RTI', '111161');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311131', 27, 'N',TO_DATE('14/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'TI', '111161');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311132', 29, 'N',TO_DATE('23/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'OCEAC', '111160');
```

```
INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311133', 30, 'N',TO_DATE('19/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'ENED', '111160');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311134', 30, 'N',TO_DATE('28/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'AEA', '111172');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311135', 26, 'N',TO_DATE('28/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'BPAS', '111172');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311136', 27, 'N', TO_DATE('29/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'METEO', '111165');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311137', 30, 'N',TO_DATE('25/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'GSE', '111151');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311138', 26, 'N',TO_DATE('18/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'AI', '111151');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311139', 26, 'N',TO_DATE('30/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'ASD', '111153');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311140', 28, 'N',TO_DATE('15/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'BIOC', '111153');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311141', 29, 'N',TO_DATE('30/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'END', '111155');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311142', 29, 'N',TO_DATE('30/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'NEU', '111155');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311143', 30, 'N',TO_DATE('30/06/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'FISGE', '111163');

INSERT INTO ESAME_SUPERATO(NUMERO_VERBALE, VOTO, LODE, DATA_ESAME, ANNO_ACCADEMICO,
CODICE_INSEGNAMENTO, MATRICOLA_STUDENTE) VALUES
('000000000311144', 26, 'N',TO_DATE('15/07/2020','DD/MM/YYYY'), TO_DATE('2019','YYYY'),
'MMSD', '111163');
```

Appello Laurea

```
INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('01/11/2020', 'DD/MM/YYYY'), '0124', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);
```

```
INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('03/11/2020', 'DD/MM/YYYY'), '0332', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('05/11/2020', 'DD/MM/YYYY'), '0328', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('02/11/2020', 'DD/MM/YYYY'), '0327', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('11/11/2020', 'DD/MM/YYYY'), '0123', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('12/11/2020', 'DD/MM/YYYY'), '0512', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('03/11/2020', 'DD/MM/YYYY'), '0122', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('09/11/2020', 'DD/MM/YYYY'), '0120', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('10/11/2020', 'DD/MM/YYYY'), '0331', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('08/11/2020', 'DD/MM/YYYY'), '0330', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('11/11/2020', 'DD/MM/YYYY'), '0326', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('07/11/2020', 'DD/MM/YYYY'), '0515', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('02/11/2020', 'DD/MM/YYYY'), '0121', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);

INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('05/11/2020', 'DD/MM/YYYY'), '0514', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);
```

```
INSERT INTO APPELLO_LAUREA(DATA_APPELLO, CODICE_CORSO, INIZIO_ISCRIZIONI, FINE_ISCRIZIONI,
TIPO, MAX_ISCRIZIONI) VALUES
(TO_DATE('01/11/2020', 'DD/MM/YYYY'), '0125', TO_DATE('01/10/2020', 'DD/MM/YYYY'),
TO_DATE('30/10/2020', 'DD/MM/YYYY'), 'PRESENZA', 5);
```

Partecipa Seduta

```
INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111112', TO_DATE('01/11/2020', 'DD/MM/YYYY'), '0124', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111126', TO_DATE('03/11/2020', 'DD/MM/YYYY'), '0332', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111125', TO_DATE('05/11/2020', 'DD/MM/YYYY'), '0328', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111115', TO_DATE('02/11/2020', 'DD/MM/YYYY'), '0327', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111115', TO_DATE('11/11/2020', 'DD/MM/YYYY'), '0123', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111121', TO_DATE('12/11/2020', 'DD/MM/YYYY'), '0512', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111118', TO_DATE('03/11/2020', 'DD/MM/YYYY'), '0122', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111119', TO_DATE('09/11/2020', 'DD/MM/YYYY'), '0120', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111124', TO_DATE('10/11/2020', 'DD/MM/YYYY'), '0331', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111128', TO_DATE('08/11/2020', 'DD/MM/YYYY'), '0330', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111119', TO_DATE('11/11/2020', 'DD/MM/YYYY'), '0326', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111129', TO_DATE('07/11/2020', 'DD/MM/YYYY'), '0515', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111122', TO_DATE('02/11/2020', 'DD/MM/YYYY'), '0121', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111120', TO_DATE('05/11/2020', 'DD/MM/YYYY'), '0514', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111111', TO_DATE('01/11/2020', 'DD/MM/YYYY'), '0125', 'Y');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111113', TO_DATE('01/11/2020', 'DD/MM/YYYY'), '0124', 'N');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111119', TO_DATE('01/11/2020', 'DD/MM/YYYY'), '0125', 'N');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111112', TO_DATE('03/11/2020', 'DD/MM/YYYY'), '0332', 'N');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111122', TO_DATE('05/11/2020', 'DD/MM/YYYY'), '0328', 'N');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111129', TO_DATE('02/11/2020', 'DD/MM/YYYY'), '0327', 'N');
```

```
INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111111', TO_DATE('11/11/2020', 'DD/MM/YYYY'), '0123', 'N');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111120', TO_DATE('12/11/2020', 'DD/MM/YYYY'), '0512', 'N');

INSERT INTO PARTECIPA_SEDUTA(TESSERINO_DOCENTE, DATA_APPELLO, CODICE_CORSO, PRESIDENTE)
VALUES ('111124', TO_DATE('03/11/2020', 'DD/MM/YYYY'), '0122', 'N');
```

Relatore

```
INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111150', '111112', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'Realizzazione di una piattaforma per l''erogazione di open data',
'SPERIMENTALE');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111151', '111126', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'Rischio idrogeologico e cambiamenti climatici: l''azione antropica
sull''ambiente', 'COMPILATIVA');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111152', '111125', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'Adaptive decision making: un''opportunità per le organizzazioni moderne',
'COMPILATIVA');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111153', '111115', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'Monitoring the Sustainability of Multi-Component Applications',
'SPERIMENTALE');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111166', '111115', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'Processi di biorisanamento ambientale', 'SPERIMENTALE');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111155', '111121', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'SPORT E DISABILITA''. IL BASKET IN CARROZZINA. ', 'COMPILATIVA');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111156', '111118', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'Allineamenti adattativi basati su elementi a microstriscia',
'COMPILATIVA');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111159', '111119', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), '
Analisi dell’interazione farmaco-cellula tramite microspettroscopia FTIR', 'SPERIMENTALE');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111160', '111124', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'Unity: un ambiente per lo sviluppo di videogiochi multipiattaforma.',
'COMPILATIVA');
```

```
INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111161', '111128', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'La green supply chain nel settore edile', 'SPERIMENTALE');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111162', '111119', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'CYBERSECURITY E POLIZIA DI STATO', 'COMPILATIVA');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111172', '111129', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'L''APPRENDIMENTO MOTORIO NEL CALCIO: LA TEORIA DEI SISTEMI DINAMICI.',
'COMPILATIVA');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111167', '111122', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'Strumenti della navigazione stimata nella storia delle Scienze Nautiche',
'COMPILATIVA');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111163', '111120', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'IGIENE E PROMOZIONE DELLA SALUTE', 'COMPILATIVA');

INSERT INTO RELATORE(MATRICOLA_STUDENTE, TESSERINO_DOCENTE, DATA_INIZIO, DATA_FINE,
TITOLO_TESI, TIPO_TESI) VALUES
('111165', '111111', TO_DATE('01/09/2020', 'DD/MM/YYYY'), TO_DATE('30/10/2020',
'DD/MM/YYYY'), 'I trasporti marittimi: organizzazione e gestione dell''attività',
'COMPILATIVA');
```

Prenotazione Appello Seduta

```
INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111150', '0124', TO_DATE('01/11/2020', 'DD/MM/YYYY'), TO_DATE('05/10/2020',
'DD/MM/YYYY'), '66601');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111151', '0332', TO_DATE('03/11/2020', 'DD/MM/YYYY'), TO_DATE('08/10/2020',
'DD/MM/YYYY'), '66602');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111152', '0328', TO_DATE('05/11/2020', 'DD/MM/YYYY'), TO_DATE('04/10/2020',
'DD/MM/YYYY'), '66603');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111153', '0327', TO_DATE('02/11/2020', 'DD/MM/YYYY'), TO_DATE('12/10/2020',
'DD/MM/YYYY'), '66604');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111166', '0123', TO_DATE('11/11/2020', 'DD/MM/YYYY'), TO_DATE('15/10/2020',
'DD/MM/YYYY'), '66605');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111155', '0512', TO_DATE('12/11/2020', 'DD/MM/YYYY'), TO_DATE('17/10/2020',
'DD/MM/YYYY'), '66606');
```

```
INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111156', '0122', TO_DATE('03/11/2020', 'DD/MM/YYYY'), TO_DATE('11/10/2020',
'DD/MM/YYYY'), '66607');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111159', '0120', TO_DATE('09/11/2020', 'DD/MM/YYYY'), TO_DATE('06/10/2020',
'DD/MM/YYYY'), '66608');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111160', '0331', TO_DATE('10/11/2020', 'DD/MM/YYYY'), TO_DATE('07/10/2020',
'DD/MM/YYYY'), '66609');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111161', '0330', TO_DATE('08/11/2020', 'DD/MM/YYYY'), TO_DATE('09/10/2020',
'DD/MM/YYYY'), '66610');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111162', '0326', TO_DATE('11/11/2020', 'DD/MM/YYYY'), TO_DATE('05/10/2020',
'DD/MM/YYYY'), '66611');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111172', '0515', TO_DATE('07/11/2020', 'DD/MM/YYYY'), TO_DATE('22/10/2020',
'DD/MM/YYYY'), '66612');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111167', '0121', TO_DATE('02/11/2020', 'DD/MM/YYYY'), TO_DATE('15/10/2020',
'DD/MM/YYYY'), '66613');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111163', '0514', TO_DATE('05/11/2020', 'DD/MM/YYYY'), TO_DATE('19/10/2020',
'DD/MM/YYYY'), '66614');

INSERT INTO PRENOTAZIONE_APPELLO_SEDUTA(MATRICOLA_STUDENTE, CODICE_CORSO, DATA_APPELLO,
DATA_PRENOTAZIONE, NUMERO) VALUES
('111165', '0125', TO_DATE('01/11/2020', 'DD/MM/YYYY'), TO_DATE('02/10/2020',
'DD/MM/YYYY'), '66615');
```

Seduta Laurea

```
INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000001','0124', TO_DATE('01/11/2020', 'DD/MM/YYYY'), '111150', 110, 'Y');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000002','0332', TO_DATE('03/11/2020', 'DD/MM/YYYY'), '111151', 90, 'N');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000003', '0328', TO_DATE('05/11/2020', 'DD/MM/YYYY'), '111152', 100, 'N');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000004','0327', TO_DATE('02/11/2020', 'DD/MM/YYYY'), '111153',  110, 'Y');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000005','0123', TO_DATE('11/11/2020', 'DD/MM/YYYY'),'111166', 100, 'N');
```

```
INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000006','0512', TO_DATE('12/11/2020', 'DD/MM/YYYY'), '111155', 110, 'Y');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000007','0122', TO_DATE('03/11/2020', 'DD/MM/YYYY'), '111156', 80, 'N');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000008','0120', TO_DATE('09/11/2020', 'DD/MM/YYYY'), '111159', 110, 'Y');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000009','0331', TO_DATE('10/11/2020', 'DD/MM/YYYY'), '111160', 110, 'Y');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000010','0330', TO_DATE('08/11/2020', 'DD/MM/YYYY'), '111161', 110, 'Y');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000011','0326', TO_DATE('11/11/2020', 'DD/MM/YYYY'), '111162', 110, 'Y');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000012','0515', TO_DATE('07/11/2020', 'DD/MM/YYYY'), '111172',  95, 'N');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000013','0121', TO_DATE('02/11/2020', 'DD/MM/YYYY'), '111167', 98, 'N');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000014','0514', TO_DATE('05/11/2020', 'DD/MM/YYYY'), '111163', 80, 'N');

INSERT INTO SEDUTA_LAUREA(NUMERO_VERBALE, CODICE_CORSO, DATA_SEDUTA, MATRICOLA_STUDENTE,
VOTO, LODE) VALUES
('400000000000015','0125', TO_DATE('01/11/2020', 'DD/MM/YYYY'), '111165', 110, 'Y');
```

Telefono Studente

```
INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111111', '111150');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111112', '111151');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111113', '111152');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111114', '111153');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111115', '111154');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111116', '111155');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111117', '111156');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111118', '111157');
```

```
INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111120', '111159');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111121', '111160');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111122', '111161');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111123', '111162');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111124', '111163');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111125', '111164');

INSERT INTO TELEFONO_STUDENTE(NUMERO_TELEFONO_STUDENTE, MATRICOLA_STUDENTE) VALUES
('3311111119', '111165');
```

Email Studente

```
INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('mario.cicalone001@studenti.uniparthenope.it', '111150');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('maria.bianchi001@studenti.uniparthenope.it', '111151');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('andrea.rossi001@studenti.uniparthenope.it', '111152');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('anna.brusaferro001@studenti.uniparthenope.it', '111153');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('luca.longhi001@studenti.uniparthenope.it', '111154');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('laura.fossetta001@studenti.uniparthenope.it', '111155');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('matteo.tromba001@studenti.uniparthenope.it', '111156');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('mario.stroppa001@studenti.uniparthenope.it', '111157');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('giuseppe.conti001@studenti.uniparthenope.it', '111159');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('camilla.nola001@studenti.uniparthenope.it', '111160');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('maria.broccoli001@studenti.uniparthenope.it', '111161');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('vincenzo.motta001@studenti.uniparthenope.it', '111162');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('carlo.mascara001@studenti.uniparthenope.it', '111163');

INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('lorenzo.manzo001@studenti.uniparthenope.it', '111164');
```

```
INSERT INTO EMAIL_STUDENTE(MAIL_STUDENTE, MATRICOLA_STUDENTE) VALUES
('anna.manita001@studenti.uniparthenope.it', '111165');
```

Telefono Docente

```
INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111111', '111111');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111112', '111112');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111113', '111113');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111114', '111114');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111115', '111115');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111116', '111116');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111117', '111117');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111118', '111118');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111119', '111119');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111120', '111120');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111121', '111121');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111122', '111122');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111123', '111123');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111124', '111124');

INSERT INTO TELEFONO_DOCENTE(NUMERO_TELEFONO_DOCENTE, TESSERINO_DOCENTE) VALUES
('3312111125', '111125');
```

Email Docente

```
INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('massimiliano.rossi@uniparthenope.it', '111111');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('giorgio.bianchi@uniparthenope.it', '111112');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('mario.verdi@uniparthenope.it', '111113');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('mariarosaria.casimeni@uniparthenope.it', '111114');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('maria.cavino@uniparthenope.it', '111115');
```

```
INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('alessio.somma@uniparthenope.it', '111116');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('daniele.siciliano@uniparthenope.it', '111117');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('gennaro.esposito@uniparthenope.it', '111118');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('manila.lanzi@uniparthenope.it', '111119');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('simone.laurenti@uniparthenope.it', '111120');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('alfredo.bengaloni@uniparthenope.it', '111121');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('francesco.amico@uniparthenope.it', '111122');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('chiara.nasone@uniparthenope.it', '111123');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('andrea.gallo@uniparthenope.it', '111124');

INSERT INTO EMAIL_DOCENTE(MAIL_DOCENTE, TESSERINO_DOCENTE) VALUES
('camillo.benso@uniparthenope.it', '111125');
```

Telefono Tutor Aziendale

```
INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111111', '111201');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111112', '111202');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111113', '111203');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111114', '111204');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111115', '111205');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111116', '111206');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111117', '111207');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111118', '111208');
```

```
INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111119', '111209');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111120', '111210');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111121', '111211');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111122', '111212');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111123', '111213');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111124', '111214');

INSERT INTO TELEFONO_TUTOR_AZIENDALE(NUMERO_TELEFONO_TUTOR_AZIENDALE,
TESSERINO_TUTOR_AZIENDA) VALUES
('3313111125', '111215');
```

Email Tutor Aziendale

```
INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('luca.bolle@uniparthenope.it', '111201');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('matteo.borghi@uniparthenope.it', '111202');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('massimo.galliotti@uniparthenope.it', '111203');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('daniele.curti@uniparthenope.it', '111204');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('angelo.posa@uniparthenope.it', '111205');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('andrea.marini@uniparthenope.it', '111206');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('franco.monte@uniparthenope.it', '111207');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('santo.bosco@uniparthenope.it', '111208');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('dario.monti@uniparthenope.it', '111209');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('marialuisa.bosco@uniparthenope.it', '111210');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('gianni.coda@uniparthenope.it', '111211');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('salvatore.trotta@uniparthenope.it', '111212');
```

```
INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('giusy.marino@uniparthenope.it', '111213');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('giovanna.sottile@uniparthenope.it', '111214');

INSERT INTO EMAIL_TUTOR_AZIENDALE(MAIL_TUTOR_AZIENDALE, TESSERINO_TUTOR_AZIENDA) VALUES
('santa.falchi@uniparthenope.it', '111215');
```

Seminario

```
INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('30/09/2019', 'DD/MM/YYYY'), '111120', 'CODING THEORY', 1, 200);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('15/10/2019', 'DD/MM/YYYY'), '111113', 'PROJECT MANAGMENT', 2, 100);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('18/10/2019','DD/MM/YYYY'), '111119', 'TECHSTARS', 3, 100);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('10/11/2019','DD/MM/YYYY'), '111111', 'BUSINESS VALUE', 2, 100);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('30/11/2019', 'DD/MM/YYYY'), '111117', 'ELIS', 3, 500);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('01/12/2019', 'DD/MM/YYYY'), '111115', 'PRODUCT PROBLEMS', 2, 100);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('07/12/2019','DD/MM/YYYY'), '111124', 'DESIGN ANALYSIS', 2, 100);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('21/02/2020', 'DD/MM/YYYY'), '111114', 'SOCIAL CORPORATE', 3, 200);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('16/03/2020', 'DD/MM/YYYY'), '111116', 'JOB PLACEMENT', 1, 200);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('21/03/2020','DD/MM/YYYY'), '111122', 'CRIPTOGRAPHY', 1, 100);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('02/04/2020','DD/MM/YYYY'), '111121', 'TRY THE FUTURE', 1, 100);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES(TO_DATE('21/04/2020', 'DD/MM/YYYY'), '111111', 'GOVERNANCE', 2, 100);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES(TO_DATE('03/05/2020', 'DD/MM/YYYY'), '111112', 'JOB TAESERUM', 3, 200);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('05/06/2020', 'DD/MM/YYYY'), '111118', 'DIRITTI WEB', 2, 100);

INSERT INTO SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, NOME, CFU, MAX_PERSONE)
VALUES (TO_DATE('15/06/2020', 'DD/MM/YYYY'), '111123', 'ETICA CONTROLLO', 1, 100);
```

Partecipa Seminario

```
INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('30/09/2019', 'DD/MM/YYYY'), '111120', '111150');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('30/09/2019', 'DD/MM/YYYY'), '111120', '111152');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('30/09/2019', 'DD/MM/YYYY'), '111120', '111161');
```

```
INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('15/10/2019', 'DD/MM/YYYY'), '111113', '111150');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('15/10/2019', 'DD/MM/YYYY'), '111113', '111153');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('18/10/2019', 'DD/MM/YYYY'), '111119', '111153');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('18/10/2019', 'DD/MM/YYYY'), '111119', '111164');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('18/10/2019', 'DD/MM/YYYY'), '111119', '111160');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('18/10/2019', 'DD/MM/YYYY'), '111119', '111159');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('30/11/2019', 'DD/MM/YYYY'), '111117', '111159');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('30/11/2019', 'DD/MM/YYYY'), '111117', '111160');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('30/11/2019', 'DD/MM/YYYY'), '111117', '111161');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('01/12/2019', 'DD/MM/YYYY'), '111115', '111162');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('01/12/2019', 'DD/MM/YYYY'), '111115', '111163');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('01/12/2019', 'DD/MM/YYYY'), '111115', '111164');

INSERT INTO PARTECIPA_SEMINARIO(DATA_SEMINARIO, TESSERINO_DOCENTE, MATRICOLA_STUDENTE)
VALUES (TO_DATE('01/12/2019', 'DD/MM/YYYY'), '111115', '111159');
```

Tassa

```
INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555535', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111150',
TO_DATE('20/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555535', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555536', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111151',
TO_DATE('20/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555536', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555537', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111152',
TO_DATE('20/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555537', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555538', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111153',
TO_DATE('20/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555538', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555539', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111154',
TO_DATE('20/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555539', 'ISCRIZIONE');
```

```
INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555540', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111155',
TO_DATE('20/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555540', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555541', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111156',
TO_DATE('20/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555541', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555542', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111157',
TO_DATE('20/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555542', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555543', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111159',
TO_DATE('21/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555543', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555544', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111160',
TO_DATE('21/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555544', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555545', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111161',
TO_DATE('22/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555545', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555546', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111162',
TO_DATE('23/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555546', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555547', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111163',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555547', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555548', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111164',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555548', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555549', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111165',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555549', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555550', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111166',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555550', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555551', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111167',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555551', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555552', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111168',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555552', 'ISCRIZIONE');
```

```
INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555553', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111169',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555553', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555554', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111170',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555554', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555555', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111171',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555555', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555556', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111172',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555556', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555557', TO_DATE('25/09/2019', 'DD/MM/YYYY'), 16, '111173',
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555557', 'ISCRIZIONE');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555558', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111150',
TO_DATE('25/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555558', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555559', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111151',
TO_DATE('26/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555559', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555560', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111152',
TO_DATE('27/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555560', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555561', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111153',
TO_DATE('28/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555561', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555562', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111154',
TO_DATE('30/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555562', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555563', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111155',
TO_DATE('30/09/2019', 'DD/MM/YYYY'), 0, '55555555555555555555563', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555564', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111156',
TO_DATE('02/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555564', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555565', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111157',
TO_DATE('03/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555565', 'PRIMA RATA');
```

```
INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555566', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111159',
TO_DATE('05/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555566', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555567', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111160',
TO_DATE('06/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555567', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555568', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111161',
TO_DATE('07/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555568', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555569', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111162',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555569', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555570', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111163',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555570', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555571', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111164',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555571', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555572', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111165',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555572', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555573', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111166',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555573', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555574', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111167',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555574', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555575', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111168',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555575', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555576', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111169',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555576', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555577', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111170',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555577', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555578', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111171',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555578', 'PRIMA RATA');
```

```
INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555579', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111172',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555579', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555580', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111173',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555580', 'PRIMA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555581', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111150',
TO_DATE('17/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555581', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555582', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111151',
TO_DATE('20/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555582', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555583', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111152',
TO_DATE('01/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555583', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555584', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111153',
TO_DATE('02/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555584', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555585', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111154',
TO_DATE('03/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555585', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555586', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111155',
TO_DATE('04/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555586', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555587', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111156',
TO_DATE('05/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555587', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555588', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111157',
TO_DATE('06/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555588', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555589', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111159',
TO_DATE('08/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555589', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555590', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111160',
TO_DATE('09/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555590', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555591', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111161',
TO_DATE('10/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555591', 'SECONDA RATA');
```

```
INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555592', TO_DATE('18/12/2019' , 'DD/MM/YYYY'), 331, '111162',
TO_DATE('19/12/2019', 'DD/MM/YYYY'), 10, '55555555555555555555592', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555593', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111163',
TO_DATE('20/12/2019', 'DD/MM/YYYY'), 10, '55555555555555555555593', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555594', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111164',
TO_DATE('29/12/2019', 'DD/MM/YYYY'), 20, '55555555555555555555594', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555595', TO_DATE('10/10/2019', 'DD/MM/YYYY'), 331, '111165',
TO_DATE('10/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555595', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555596', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111166',
TO_DATE('10/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555596', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555597', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111167',
TO_DATE('10/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555597', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555598', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111168',
TO_DATE('10/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555598', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555599', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111169',
TO_DATE('10/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555599', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555600', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111170',
TO_DATE('08/10/2019', 'DD/MM/YYYY'), 0, '55555555555555555555600', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555601', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111171',
TO_DATE('10/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555601', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555602', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111172',
TO_DATE('10/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555602', 'SECONDA RATA');

INSERT INTO TASSA(NUMERO_FATTURA, SCADENZA, IMPORTO, MATRICOLA_STUDENTE, DATA_PAGAMENTO,
MORA, IUV, TIPO)
VALUES ('555555555555603', TO_DATE('18/12/2019', 'DD/MM/YYYY'), 331, '111173',
TO_DATE('10/11/2019', 'DD/MM/YYYY'), 0, '55555555555555555555603', 'SECONDA RATA');
```

Ricevimento

```
INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('01/12/2019 15:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111111', 120);
```

```
INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('01/12/2019 17:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111111', 60);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('01/12/2019 18:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111111', 90);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('08/12/2019 14:30:00', 'DD/MM/YYYY HH24:MI:SS'), '111112', 90);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('08/12/2019 16:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111112', 60);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('08/12/2019 17:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111112', 90);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('12/12/2019 15:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111113', 120);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('12/12/2019 17:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111113', 30);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('12/12/2019 17:30:00', 'DD/MM/YYYY HH24:MI:SS'), '111113', 90);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('14/12/2019 15:30:00', 'DD/MM/YYYY HH24:MI:SS'), '111114', 90);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('14/12/2019 17:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111114', 120);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('14/12/2019 19:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111114', 30);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('19/12/2019 14:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111115', 120);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('19/12/2019 16:00:00', 'DD/MM/YYYY HH24:MI:SS'), '111115', 90);

INSERT INTO RICEVIMENTO(DATA_RICEVIMENTO, TESSERINO_DOCENTE, DURATA)
VALUES (TO_DATE('19/12/2019 17:30:00', 'DD/MM/YYYY HH24:MI:SS'), '111115', 90);
```

Prenotazione Ricevimento

```
INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000001', TO_DATE('01/12/2019 15:00:00', 'DD/MM/YYYY
HH24:MI:SS'),'111111', '111150', TO_DATE('01/12/2019 09:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000002', TO_DATE('01/12/2019 17:00:00', 'DD/MM/YYYY
HH24:MI:SS'),'111111', '111151', TO_DATE('01/12/2019 10:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000003', TO_DATE('01/12/2019 18:00:00', 'DD/MM/YYYY
HH24:MI:SS'),'111111', '111152', TO_DATE('01/12/2019 10:30:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000004', TO_DATE('08/12/2019 14:30:00', 'DD/MM/YYYY HH24:MI:SS'),
'111112', '111150', TO_DATE('08/12/2019 09:00:00', 'DD/MM/YYYY HH24:MI:SS'));
```

```
INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000005', TO_DATE('08/12/2019 16:00:00', 'DD/MM/YYYY HH24:MI:SS'),
'111112', '111151', TO_DATE('08/12/2019 10:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000006', TO_DATE('08/12/2019 17:00:00', 'DD/MM/YYYY HH24:MI:SS'),
'111112', '111152', TO_DATE('08/12/2019 10:30:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000007', TO_DATE('12/12/2019 15:00:00', 'DD/MM/YYYY HH24:MI:SS'),
'111113', '111153', TO_DATE('12/12/2019 09:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000008', TO_DATE('12/12/2019 17:00:00', 'DD/MM/YYYY
HH24:MI:SS'),'111113', '111154', TO_DATE('12/12/2019 10:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000009', TO_DATE('12/12/2019 17:30:00', 'DD/MM/YYYY HH24:MI:SS'),
'111113', '111155', TO_DATE('12/12/2019 10:30:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000010', TO_DATE('14/12/2019 15:30:00', 'DD/MM/YYYY HH24:MI:SS'),
'111114', '111156', TO_DATE('12/12/2019 09:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000011', TO_DATE('14/12/2019 17:00:00', 'DD/MM/YYYY HH24:MI:SS'),
'111114', '111154', TO_DATE('14/12/2019 09:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000012', TO_DATE('14/12/2019 19:00:00', 'DD/MM/YYYY HH24:MI:SS'),
'111114', '111155', TO_DATE('14/12/2019 09:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000013', TO_DATE('19/12/2019 14:00:00', 'DD/MM/YYYY HH24:MI:SS'),
'111115', '111150', TO_DATE('19/12/2019 09:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000014', TO_DATE('19/12/2019 16:00:00', 'DD/MM/YYYY HH24:MI:SS'),
'111115', '111151', TO_DATE('19/12/2019 10:00:00', 'DD/MM/YYYY HH24:MI:SS'));

INSERT INTO PRENOTAZIONE_RICEVIMENTO(NUMERO_PRENOTAZIONE, DATA_RICEVIMENTO,
TESSERINO_DOCENTE, MATRICOLA_STUDENTE, DATA_PRENOTAZIONE)
VALUES ('000000000000015', TO_DATE('19/12/2019 17:30:00', 'DD/MM/YYYY HH24:MI:SS'),
'111115', '111152', TO_DATE('19/12/2019 10:30:00', 'DD/MM/YYYY HH24:MI:SS'));
```

Bando Erasmus

```
INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555580', TO_DATE('01/06/2020', 'DD/MM/YYYY'),
TO_DATE('01/03/2019', 'DD/MM/YYYY'), 60, 200);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555581', TO_DATE('02/06/2020', 'DD/MM/YYYY'),
TO_DATE('02/03/2019', 'DD/MM/YYYY'), 60, 200);
```

```
INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555582', TO_DATE('03/06/2020', 'DD/MM/YYYY'),
TO_DATE('03/03/2019', 'DD/MM/YYYY'), 60, 200);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555583', TO_DATE('04/06/2020', 'DD/MM/YYYY'),
TO_DATE('04/03/2019', 'DD/MM/YYYY'), 30, 150);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555584', TO_DATE('05/06/2020', 'DD/MM/YYYY'),
TO_DATE('05/03/2019', 'DD/MM/YYYY'), 30, 150);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555585', TO_DATE('06/06/2020', 'DD/MM/YYYY'),
TO_DATE('06/03/2019', 'DD/MM/YYYY'), 30, 150);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555586', TO_DATE('07/06/2020', 'DD/MM/YYYY'),
TO_DATE('07/03/2019', 'DD/MM/YYYY'), 60, 200);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555587', TO_DATE('08/06/2020', 'DD/MM/YYYY'),
TO_DATE('08/03/2019', 'DD/MM/YYYY'), 30, 150);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555588', TO_DATE('09/06/2020', 'DD/MM/YYYY'),
TO_DATE('09/03/2019', 'DD/MM/YYYY'), 60, 200);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555589', TO_DATE('10/06/2020', 'DD/MM/YYYY'),
TO_DATE('10/03/2019', 'DD/MM/YYYY'), 60, 200);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555590', TO_DATE('11/06/2020', 'DD/MM/YYYY'),
TO_DATE('11/03/2019', 'DD/MM/YYYY'), 60, 200);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555591', TO_DATE('12/06/2020', 'DD/MM/YYYY'),
TO_DATE('12/03/2019', 'DD/MM/YYYY'), 60, 200);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555592', TO_DATE('13/06/2020', 'DD/MM/YYYY'),
TO_DATE('13/03/2019', 'DD/MM/YYYY'), 30, 150);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555593', TO_DATE('14/06/2020', 'DD/MM/YYYY'),
TO_DATE('14/03/2019', 'DD/MM/YYYY'), 30, 150);

INSERT INTO BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, SCADENZA, DATA_EMISSIONE, CFU,
NUMERO_POSTI) VALUES ('555555555555594', TO_DATE('15/06/2020', 'DD/MM/YYYY'),
TO_DATE('15/03/2019', 'DD/MM/YYYY'), 30, 150);
```

Partecipazione Bando Erasmus

```
INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555580', '111150', TO_DATE('01/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555580', '111151', TO_DATE('02/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555580', '111152', TO_DATE('03/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555580', '111153', TO_DATE('04/11/2019', 'DD/MM/YYYY'));
```

```
INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555580', '111154', TO_DATE('05/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555580', '111155', TO_DATE('06/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555581', '111156', TO_DATE('05/12/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555581', '111157', TO_DATE('06/12/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555581', '111170', TO_DATE('07/12/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555582', '111159', TO_DATE('08/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555582', '111160', TO_DATE('09/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555582', '111161', TO_DATE('10/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555582', '111162', TO_DATE('11/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555582', '111163', TO_DATE('12/11/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555582', '111164', TO_DATE('13/11/2019', 'DD/MM/YYYY'));
```

Assegnazione Erasmus

```
INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555580', '111150', TO_DATE('01/06/2019', 'DD/MM/YYYY'), 'CADICE',
'UNIVERSIDAD DE CADIZ', TO_DATE('10/09/2019', 'DD/MM/YYYY'), TO_DATE('10/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555580', '111151', TO_DATE('02/06/2019', 'DD/MM/YYYY'), 'CADICE',
'UNIVERSIDAD DE CADIZ', TO_DATE('11/09/2019', 'DD/MM/YYYY'), TO_DATE('11/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555580', '111152', TO_DATE('03/06/2019', 'DD/MM/YYYY'), 'CADICE',
'UNIVERSIDAD DE CADIZ', TO_DATE('12/09/2019', 'DD/MM/YYYY'), TO_DATE('12/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555580', '111153', TO_DATE('04/06/2019', 'DD/MM/YYYY'), 'CADICE',
'UNIVERSIDAD DE CADIZ', TO_DATE('13/09/2019', 'DD/MM/YYYY'), TO_DATE('13/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555580', '111154', TO_DATE('05/06/2019', 'DD/MM/YYYY'), 'CADICE',
'UNIVERSIDAD DE CADIZ', TO_DATE('14/09/2019', 'DD/MM/YYYY'), TO_DATE('14/02/2020',
'DD/MM/YYYY'));
```

```
INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555580', '111155', TO_DATE('06/06/2019', 'DD/MM/YYYY'), 'CADICE',
'UNIVERSIDAD DE CADIZ', TO_DATE('15/09/2019', 'DD/MM/YYYY'), TO_DATE('15/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555581', '111156', TO_DATE('05/07/2019', 'DD/MM/YYYY'), 'LAS PALMAS',
'UNIVERSIDAD DE PALMAS', TO_DATE('10/09/2019', 'DD/MM/YYYY'), TO_DATE('10/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555581', '111157', TO_DATE('06/07/2019', 'DD/MM/YYYY'), 'LAS PALMAS',
'UNIVERSIDAD DE PALMAS', TO_DATE('11/09/2019', 'DD/MM/YYYY'), TO_DATE('11/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555581', '111170', TO_DATE('07/07/2019', 'DD/MM/YYYY'), 'LAS PALMAS',
'UNIVERSIDAD DE PALMAS', TO_DATE('12/09/2019', 'DD/MM/YYYY'), TO_DATE('12/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555582', '111159', TO_DATE('08/07/2019', 'DD/MM/YYYY'), 'SIVIGLIA',
'UNIVERSIDAD DE SEVILLA', TO_DATE('10/09/2019', 'DD/MM/YYYY'), TO_DATE('10/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555582', '111160', TO_DATE('09/07/2019', 'DD/MM/YYYY'), 'SIVIGLIA',
'UNIVERSIDAD DE SEVILLA', TO_DATE('11/09/2019', 'DD/MM/YYYY'), TO_DATE('11/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555582', '111161', TO_DATE('10/07/2019', 'DD/MM/YYYY'), 'SIVIGLIA',
'UNIVERSIDAD DE SEVILLA', TO_DATE('12/09/2019', 'DD/MM/YYYY'), TO_DATE('12/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555582', '111162', TO_DATE('11/07/2019', 'DD/MM/YYYY'), 'SIVIGLIA',
'UNIVERSIDAD DE SEVILLA', TO_DATE('13/09/2019', 'DD/MM/YYYY'), TO_DATE('13/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555582', '111163', TO_DATE('12/07/2019', 'DD/MM/YYYY'), 'SIVIGLIA',
'UNIVERSIDAD DE SEVILLA', TO_DATE('14/09/2019', 'DD/MM/YYYY'), TO_DATE('14/02/2020',
'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_ERASMUS(NUMERO_BANDO_ERASMUS, MATRICOLA_STUDENTE,
DATA_ASSEGNAZIONE, LOCALITA, NOME_UNIVERSITA, DATA_PARTENZA, DATA_RIENTRO)
VALUES ('555555555555582', '111164', TO_DATE('13/07/2019', 'DD/MM/YYYY'), 'SIVIGLIA',
'UNIVERSIDAD DE SEVILLA', TO_DATE('15/09/2019', 'DD/MM/YYYY'), TO_DATE('15/02/2020',
'DD/MM/YYYY'));
```

Bando Borsa

```
INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555595', TO_DATE('20/09/2020', 'DD/MM/YYYY'),
TO_DATE('20/09/2019', 'DD/MM/YYYY'), 1000, 'VINCITORE POR/FSE', 'VINCITORE FONDO POR/FSE',
1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555596', TO_DATE('21/09/2020', 'DD/MM/YYYY'),
TO_DATE('21/09/2019', 'DD/MM/YYYY'), 1000, 'MIGLIOR STUDENTE ANNO', 'FONDI ATENEO', 1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555597', TO_DATE('22/09/2020', 'DD/MM/YYYY'),
TO_DATE('22/09/2019', 'DD/MM/YYYY'), 1000, 'VINCITORE BORSA ATENEO', 'FONDI ATENEO', 1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555598', TO_DATE('23/09/2020', 'DD/MM/YYYY'),
TO_DATE('23/09/2019', 'DD/MM/YYYY'), 2000, 'MIGLIOR STUDENTE REGIONE', 'REGIONE CAMPANIA',
1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555599', TO_DATE('24/09/2020', 'DD/MM/YYYY'),
TO_DATE('24/09/2019', 'DD/MM/YYYY'), 1500, 'VINCITORE POR/FSE', 'VINCITORE FONDO POR/FSE',
1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555600', TO_DATE('25/09/2020', 'DD/MM/YYYY'),
TO_DATE('25/09/2019', 'DD/MM/YYYY'), 1500, 'MIGLIOR STUDENTE ANNO', 'FONDI ATENEO', 1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555601', TO_DATE('26/09/2020', 'DD/MM/YYYY'),
TO_DATE('26/09/2019', 'DD/MM/YYYY'), 2000, 'VINCITORE BORSA ATENEO', 'FONDI ATENEO', 1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555602', TO_DATE('27/09/2020', 'DD/MM/YYYY'),
TO_DATE('27/09/2019', 'DD/MM/YYYY'), 2500, 'MIGLIOR STUDENTE REGIONE', 'REGIONE CAMPANIA',
1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555603', TO_DATE('28/09/2020', 'DD/MM/YYYY'),
TO_DATE('28/09/2019', 'DD/MM/YYYY'), 1000, 'VINCITORE POR/FSE', 'VINCITORE FONDO POR/FSE',
1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555604', TO_DATE('29/09/2020', 'DD/MM/YYYY'),
TO_DATE('29/09/2019', 'DD/MM/YYYY'), 1000, 'MIGLIOR STUDENTE ANNO', 'FONDI ATENEO', 1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555605', TO_DATE('01/09/2021', 'DD/MM/YYYY'),
TO_DATE('01/09/2020', 'DD/MM/YYYY'), 1000, 'VINCITORE BORSA ATENEO', 'FONDI ATENEO', 1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555606', TO_DATE('02/09/2021', 'DD/MM/YYYY'),
TO_DATE('02/09/2020', 'DD/MM/YYYY'), 1000, 'MIGLIOR STUDENTE REGIONE', 'REGIONE CAMPANIA',
1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555607', TO_DATE('03/09/2021', 'DD/MM/YYYY'),
TO_DATE('03/10/2020', 'DD/MM/YYYY'), 1000, 'VINCITORE POR/FSE', 'VINCITORE FONDO POR/FSE',
1000);

INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555608', TO_DATE('04/09/2021', 'DD/MM/YYYY'),
TO_DATE('04/10/2020', 'DD/MM/YYYY'), 1000, 'VINCITORE BORSA ATENEO', 'FONDI ATENEO', 1000);
```

```
INSERT INTO BANDO_BORSA(NUMERO_BANDO_BORSA, SCADENZA, DATA_EMISSIONE, VALORE, CAUSALE,
TIPO, NUMERO_BORSE) VALUES ('555555555555609', TO_DATE('05/09/2021', 'DD/MM/YYYY'),
TO_DATE('05/10/2020', 'DD/MM/YYYY'), 1000, 'MIGLIOR STUDENTE REGIONE', 'REGIONE CAMPANIA',
1000);
```

Partecipazione Bando Borsa

```
INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555595', '111150', TO_DATE('01/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555596', '111151', TO_DATE('02/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555597', '111152', TO_DATE('03/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555598', '111153', TO_DATE('04/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555599', '111154', TO_DATE('05/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555600', '111155', TO_DATE('06/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555601', '111156', TO_DATE('05/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555602', '111157', TO_DATE('06/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555603', '111170', TO_DATE('07/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555604', '111159', TO_DATE('08/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555605', '111160', TO_DATE('09/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555606', '111161', TO_DATE('10/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555607', '111162', TO_DATE('11/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555608', '111163', TO_DATE('12/10/2019', 'DD/MM/YYYY'));

INSERT INTO PARTECIPAZIONE_BANDO_BORSA(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE,
DATA_DOMANDA) VALUES ('555555555555609', '111164', TO_DATE('13/10/2019', 'DD/MM/YYYY'));
```

Assegnazione Borse

```
INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555595', '111150', TO_DATE('01/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555596', '111151', TO_DATE('02/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555597', '111152', TO_DATE('03/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555598', '111153', TO_DATE('04/11/2019', 'DD/MM/YYYY'));
```

```
INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555599', '111154', TO_DATE('05/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555600', '111155', TO_DATE('06/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555601', '111156', TO_DATE('05/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555602', '111157', TO_DATE('06/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555603', '111170', TO_DATE('07/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555604', '111159', TO_DATE('08/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555605', '111160', TO_DATE('09/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555606', '111161', TO_DATE('10/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555607', '111162', TO_DATE('11/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555608', '111163', TO_DATE('12/11/2019', 'DD/MM/YYYY'));

INSERT INTO ASSEGNAZIONE_BORSE(NUMERO_BANDO_BORSA, MATRICOLA_STUDENTE, DATA_ASSEGNAZIONE)
VALUES ('555555555555609', '111164', TO_DATE('13/11/2019', 'DD/MM/YYYY'));
```

Trigger

Non tutti i tipi di vincoli sono esprimibili attraverso i comandi o le strutture viste finora. I trigger sono utili principalmente per il controllo di vincoli dinamici in fase di immissione o aggiornamento dei dati. È possibile usarli anche per implementare *regole di business* o per calcolare, generare o sovrascrivere il valore di alcuni campi. Nel caso del DB *esse4,* i trigger sono stati usati per verificare dei vincoli dinamici in fase di immissione e modellare alcuni aspetti del contesto analizzato che non possono essere controllati semplicemente dai vincoli espressi nel DDL. Di seguito si riportano i 9 trigger implementati:

Verifica prenotazione appello

Uno studente può prenotarsi ad un appello di un'edizione insegnamento se e solo se:

- l'appello riguarda il suo corso di laurea
- ha pagato tutte le tasse
- non ci sono più prenotazioni per lo stesso appello dello stesso studente
- non è stato già superato il numero di iscritti consentiti per l'appello
- non ha superato ancora quell'esame
- la prenotazione è avvenuta in una data coerente con quelle di inizio e fine prenotazione dell'appello

Il seguente trigger verifica tutte queste situazioni, per poi effettuare la prenotazione dell'appello:

```
CREATE OR REPLACE TRIGGER VERIFICA_PRENOTAZIONE_APPELLO BEFORE INSERT OR UPDATE ON
prenotazione_appello FOR EACH ROW
DECLARE
    studentID prenotazione_appello.matricola_studente % TYPE := :NEW.matricola_studente;
    studentCourseID studente.codice_corso % TYPE;

    dataPrenotazione prenotazione_appello.data_prenotazione % TYPE :=
       :NEW.data_prenotazione;
    codiceInsegnamentoAppello prenotazione_appello.codice_insegnamento % TYPE :=
       :NEW.codice_insegnamento;

    tuplaAppello appello % ROWTYPE;
    tuplaStudente studente % ROWTYPE;

    tempMatricolaStudente studente.matricola_studente % TYPE;
    nPostiPrenotati appello.max_studenti % TYPE := 0;
    studentID_payment prenotazione_appello_seduta.matricola_studente % TYPE;

    invalidBookingDate EXCEPTION;
    postiMancanti EXCEPTION;
    unpaidTaxes EXCEPTION;
BEGIN
    -- verifica correttezza CdL. Prelevo il CdL di tale studente
    SELECT studente.codice_corso INTO studentCourseID
    FROM studente
    WHERE studente.matricola_studente = studentID;

   -- verifica correttezza appello. recupero l'appello a cui tale studente vuole prenotarsi
    SELECT appello.anno_accademico, appello.data_appello, appello.codice_insegnamento,
       appello.data_inizio, appello.data_fine, appello.max_studenti, appello.tipo INTO
       tuplaAppello
    FROM appello
    WHERE appello.codice_insegnamento = :NEW.codice_insegnamento
        AND appello.anno_accademico = :NEW.anno_accademico
        AND appello.data_appello =:NEW.data_appello;
```

```
    -- verifico date. Verifico che la data di prenotazione per tale appello sia all'interno
    -- del periodo in cui è possibile prenotarsi
    IF (tuplaAppello.data_inizio > dataPrenotazione OR
       tuplaAppello.data_fine < dataPrenotazione) THEN
        RAISE invalidBookingDate;
    END IF;

    -- verifico posti. Conto il numero di persone già prenotate a tale appello
    SELECT NVL(cont, 0)
    INTO nPostiPrenotati
    FROM
       ((SELECT cont FROM
                -- recupero il numero di prenotazioni per tale appello
               (SELECT appello.data_appello, appello.codice_insegnamento,
                     appello.anno_accademico, COUNT(*) as cont
                FROM appello JOIN prenotazione_appello
                ON appello.anno_accademico = prenotazione_appello.anno_accademico
                AND appello.codice_insegnamento = prenotazione_appello.codice_insegnamento
                AND appello.data_appello = prenotazione_appello.data_appello
                GROUP BY appello.data_appello, appello.codice_insegnamento,
                     appello.anno_accademico
                HAVING appello.data_appello = tuplaAppello.data_appello
                AND appello.codice_insegnamento = tuplaAppello.codice_insegnamento
                AND appello.anno_accademico = tuplaAppello.anno_accademico))
         UNION ALL SELECT 0 FROM DUAL
       )
    WHERE ROWNUM = 1;

    -- verifico che ci siano posti disponibili per tale appello
    IF (nPostiPrenotati >= tuplaAppello.max_studenti) THEN
        RAISE postiMancanti;
    END IF;

    -- verifico CdL studente e CdL coinvolti in appello
    SELECT * INTO tuplaStudente
    FROM studente
    WHERE studente.matricola_studente = studentID
    AND EXISTS (SELECT corso_laurea.codice_corso
                FROM ((appello JOIN offerta_insegnamento
                       ON appello.codice_insegnamento =
                       offerta_insegnamento.codice_insegnamento
                       )
                       JOIN corso_laurea
                       ON corso_laurea.codice_corso = offerta_insegnamento.codice_corso)
                WHERE corso_laurea.codice_corso = studentCourseID
                AND appello.anno_accademico = tuplaAppello.anno_accademico
                AND appello.data_appello = tuplaAppello.data_appello
                AND appello.codice_insegnamento = tuplaAppello.codice_insegnamento
     );

/*
    non può prenotarsi per esami già superati: non esiste un esame superato la cui
    matricola studente è la stessa presente sulla prenotazione che si sta cercando di
    inserire
*/
    SELECT studente.matricola_studente INTO tempMatricolaStudente
    FROM studente
    WHERE studente.matricola_studente = studentID AND
    NOT EXISTS (SELECT *
                FROM esame_superato
                WHERE esame_superato.matricola_studente = studentID
                AND esame_superato.codice_insegnamento = codiceInsegnamentoAppello
               );


    -- verifica pagamento di tutte le tasse: non esistono tasse non pagate dallo studente
```

```
    SELECT studente.matricola_studente INTO studentID_payment
    FROM studente LEFT JOIN tassa ON studente.matricola_studente = tassa.matricola_studente
    WHERE (NOT EXISTS (SELECT * FROM tassa
                       WHERE tassa.matricola_studente = studentID AND
                       tassa.data_pagamento IS NULL
                      )
          ) AND studente.matricola_studente = studentID
    GROUP BY studente.matricola_studente;
    -- se vi sono tasse non pagate, tale studente non può prenotarsi
    IF (studentID_payment IS NULL) THEN
        RAISE unpaidTaxes;
    END IF;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Non puoi prenotarti per questo esame!');
    WHEN invalidBookingDate THEN
        RAISE_APPLICATION_ERROR(-20001, 'Data prenotazione non valida!');
    WHEN postiMancanti THEN
        RAISE_APPLICATION_ERROR(-20002, 'Posti Mancanti!');
    WHEN unpaidTaxes THEN
        RAISE_APPLICATION_ERROR(-20003, 'Sono presenti delle tasse non pagate. Non puoi
                                iscriverti all''appello!');
END;
```

Verifica prenotazione appello di laurea

Uno studente può effettuare una prenotazione per un appello di seduta di laurea se e solo se:

- l'appello di seduta di laurea per il quale ha eseguito la prenotazione afferisce al suo CdL
- se la somma dei CFU ottenuti con i seminari (al più 3), dei CFU ottenuti con il tirocinio (12), dei CFU ottenuti con i bandi erasmus (al più 12) e quelli ottenuti con gli esami del CdL è maggiore di 180 se si tratta di una triennale o 120 se si tratta di una magistrale
- ha pagato tutte le tasse previste
- la data di prenotazione è avvenuta nel range di date corretto
- ci sono abbastanza posti rimanenti
- ha effettuato il tirocinio

Il seguente trigger verifica tutte queste situazioni, per poi effettuare la prenotazione dell'appello di laurea:

```
CREATE OR REPLACE TRIGGER VERIFICA_PRENOTAZIONE_APPELLO_LAUREA
BEFORE INSERT OR UPDATE ON prenotazione_appello_seduta FOR EACH ROW
DECLARE
    studentID prenotazione_appello_seduta.matricola_studente % TYPE :=
       :NEW.matricola_studente;
    insertedCourseID studente.codice_corso % TYPE := :NEW.codice_corso;
    studentCourseID studente.codice_corso % TYPE;
    studentID_payment prenotazione_appello_seduta.matricola_studente % TYPE;
    studentID_temp prenotazione_appello_seduta.matricola_studente % TYPE;

    CFU_seminari seminario.CFU % TYPE := 0;
    CFU_tirocini tirocinio.CFU % TYPE := 0;
    CFU_esamiSuperati NUMBER(3, 0) := 0;
    CFU_bandiErasmus bando_erasmus.CFU % TYPE := 0;

    nPostiPrenotati appello_laurea.max_iscrizioni % TYPE := 0;
    tuplaAppello appello_laurea % ROWTYPE;
    dataPrenotazione prenotazione_appello_seduta.data_prenotazione % TYPE :=
       :NEW.data_prenotazione;

    tipoLaurea corso_laurea.tipo % TYPE := 0;
    notEnough_CFU EXCEPTION;
    invalidCourseID EXCEPTION;
```

```
    unpaidTaxes EXCEPTION;
    invalidBookingDate EXCEPTION;
    postiMancanti EXCEPTION;
BEGIN
    -- verifica correttezza CdL. Prelevo il CdL effettivo dello studente caricato nel DB
    SELECT studente.codice_corso INTO studentCourseID
    FROM studente
    WHERE studente.matricola_studente = studentID;

    IF studentCourseID != insertedCourseID THEN
        RAISE invalidCourseID;
    END IF;

    -- conteggio CFU seminari
    -- somma dei cfu dei seminari a cui ha partecipato tale studente
    SELECT studente.matricola_studente, SUM(seminario.CFU) INTO studentID_temp,
       CFU_seminari
    FROM ( /* recupero gli studenti che hanno partecipato a seminari e per tale seminari
              recupero il numero di CFU corrispondente */

       (STUDENTE LEFT JOIN partecipa_seminario ON STUDENTE.matricola_studente =
           partecipa_seminario.matricola_studente) LEFT JOIN seminario ON
           partecipa_seminario.data_seminario = seminario.data_seminario
           AND partecipa_seminario.tesserino_docente = seminario.tesserino_docente
    )
    WHERE studente.matricola_studente = studentID
    GROUP BY studente.matricola_studente;

    -- se non ha partecipato ad alcun seminario, assegno 0 a tale conteggio
    IF (CFU_seminari IS NULL) THEN
        CFU_seminari := 0;
    END IF;

    -- limito il numero di CFU per i seminari a 3 se eventualmente ha assistito a più
    -- seminari ottenendo più CFU dei massimi consentiti
    IF (CFU_seminari > 3) THEN
        CFU_seminari := 3;
    END IF;

    -- conteggio CFU tirocini
    -- prelevo il numero dei cfu del tirocinio a cui ha partecipato tale studente, se vi ha
    -- partecipato.
    SELECT tirocinio.matricola_studente, tirocinio.CFU INTO studentID_temp, CFU_tirocini
    FROM tirocinio
    WHERE tirocinio.matricola_studente = studentID;

    -- si suppone che il massimo numero di CFU ottenibili da tirocini sia 12 (vedi traccia
    -- trigger)
    IF CFU_tirocini > 12 THEN
        CFU_tirocini := 12;
    END IF;

    -- conteggio CFU Erasmus
    -- somma dei CFU conseguiti dalle esperienze Erasmus a cui ha partecipato tale studente
    SELECT studente.matricola_studente, SUM(bando_erasmus.CFU) INTO studentID_temp,
       CFU_bandiErasmus
    FROM (studente LEFT JOIN assegnazione_erasmus ON STUDENTE.matricola_studente =
         assegnazione_erasmus.matricola_studente) LEFT JOIN bando_erasmus ON
         assegnazione_erasmus.numero_bando_erasmus = bando_erasmus.numero_bando_erasmus
    WHERE studente.matricola_studente = studentID
    GROUP BY studente.matricola_studente;

    -- se non ha partecipato ad alcun erasmus, assegno 0 a tale conteggio
    IF (CFU_bandiErasmus IS NULL) THEN
        CFU_bandiErasmus := 0;
    END IF;
```

```
-- si suppone max 12 cfu per erasmus in totale
IF CFU_bandiErasmus > 12 THEN
    CFU_bandiErasmus := 12;
END IF;

-- conteggio CFU esami superati. Somma dei CFU conseguiti al superamento di esami
SELECT studente.matricola_studente, SUM(edizione_insegnamento.CFU) INTO studentID_temp,
   CFU_esamiSuperati
FROM ((
   /* recupero gli esami superati, i relativi appelli e le relative edizioni
    dell'insegnamento per prelevare il numero di CFU corrispondente */
   (studente LEFT JOIN esame_superato ON studente.matricola_studente =
        esame_superato.matricola_studente)
        LEFT JOIN appello ON esame_superato.data_esame = appello.data_appello
            AND esame_superato.anno_accademico = appello.anno_accademico
            AND esame_superato.codice_insegnamento = appello.codice_insegnamento)
        LEFT JOIN edizione_insegnamento ON edizione_insegnamento.anno_accademico =
        appello.anno_accademico AND edizione_insegnamento.codice_insegnamento =
        appello.codice_insegnamento
)
WHERE studente.matricola_studente = studentID
GROUP BY studente.matricola_studente;

-- se non ha superato alcun esame, assegno 0 a tale conteggio
IF (CFU_esamiSuperati IS NULL) THEN
    CFU_esamiSuperati := 0;
END IF;

-- verifica tipo seduta laurea. Prelevo il tipo di laurea di tale studente
SELECT corso_laurea.tipo INTO tipoLaurea
FROM studente JOIN corso_laurea ON studente.codice_corso = corso_laurea.codice_corso
WHERE studente.matricola_studente = studentID;

/*
Verifica il tipo del corso di laurea al quale è iscritto lo studente e se può
prenotarsi per un appello di seduta di laurea del proprio corso.
Se la laurea di tale studente è triennale, la somma deve essere pari a 180 mentre se la
laurea di tale studente è magistrale, la somma deve essere pari a 120
*/
IF LOWER(tipoLaurea) = 'triennale' THEN
    IF (CFU_esamiSuperati + CFU_tirocini + CFU_seminari +  CFU_bandiErasmus) < 180 THEN
        RAISE notEnough_CFU;
    END IF;
ELSIF LOWER(tipoLaurea) = 'magistrale' THEN
    IF (CFU_esamiSuperati + CFU_tirocini + CFU_seminari +  CFU_bandiErasmus) < 120 THEN
        RAISE notEnough_CFU;
    END IF;
END IF;

-- verifica pagamento di tutte le tasse: non esistono tasse non pagate dallo studente
SELECT studente.matricola_studente INTO studentID_payment
FROM studente LEFT JOIN tassa ON studente.matricola_studente = tassa.matricola_studente
WHERE (NOT EXISTS (     SELECT *
                        FROM tassa
                        WHERE tassa.matricola_studente = studentID AND
                        tassa.data_pagamento IS NULL
                  )
       ) AND studente.matricola_studente = studentID
GROUP BY studente.matricola_studente;

-- se vi sono tasse non pagate, tale studente non può prenotarsi
IF (studentID_payment IS NULL) THEN
    RAISE unpaidTaxes;
END IF;

-- prelievo appello a cui tale studente vuole prenotarsi
```

```
    SELECT appello_laurea.data_appello, appello_laurea.codice_corso,
       appello_laurea.inizio_iscrizioni, appello_laurea.fine_iscrizioni,
       appello_laurea.tipo, appello_laurea.max_iscrizioni INTO tuplaAppello
    FROM appello_laurea
    WHERE appello_laurea.codice_corso = :NEW.codice_corso AND appello_laurea.data_appello
       =:NEW.data_appello;

    -- verifico se la data di prenotazione è compresa tra l'inizio e la fine del periodo di
    -- iscrizione dell'appello
    IF (tuplaAppello.inizio_iscrizioni > dataPrenotazione OR tuplaAppello.fine_iscrizioni <
       dataPrenotazione) THEN
        RAISE invalidBookingDate;
    END IF;

    -- verifico posti. Conto il numero di persone già prenotate a tale appello
    SELECT NVL(cont, 0)
    INTO nPostiPrenotati
    FROM ((SELECT cont
              -- recupero il numero di prenotazioni per tale appello
           FROM
           (SELECT appello_laurea.data_appello, appello_laurea.codice_corso, COUNT(*) as
            cont FROM appello_laurea JOIN prenotazione_appello_seduta
            ON appello_laurea.codice_corso = prenotazione_appello_seduta.codice_corso
            AND appello_laurea.data_appello = prenotazione_appello_seduta.data_appello
            GROUP BY appello_laurea.data_appello, appello_laurea.codice_corso
            HAVING appello_laurea.data_appello = tuplaAppello.data_appello
            AND appello_laurea.codice_corso = tuplaAppello.codice_corso
            ))
            UNION ALL SELECT 0 FROM DUAL
            )
    WHERE ROWNUM = 1;

    -- verifico che ci siano posti disponibili per tale appello
    IF (nPostiPrenotati >= tuplaAppello.max_iscrizioni) THEN
        RAISE postiMancanti;
    END IF;
EXCEPTION
    WHEN notEnough_CFU THEN
        RAISE_APPLICATION_ERROR(-20010, 'Non e'' possibile effettuare la prenotazione. Non
                                hai raggiunto abbastanza CFU!');
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20011, 'Non e'' stato possibile completare l''operazione
                                di prenotazione.');
    WHEN invalidCourseID THEN
        RAISE_APPLICATION_ERROR(-20012, 'Codice di Corso di Laurea non valido');
    WHEN unpaidTaxes THEN
        RAISE_APPLICATION_ERROR(-20013, 'Sono presenti delle tasse non pagate. Non puoi
                                laurearti!');
    WHEN invalidBookingDate THEN
        RAISE_APPLICATION_ERROR(-20014, 'Data prenotazione non valida!');
    WHEN postiMancanti THEN
        RAISE_APPLICATION_ERROR(-20015, 'Posti Mancanti!');
END;
```

Verifica prenotazione bando borsa di studio

Uno studente non può effettuare la domanda di partecipazione a un bando borsa di studio prima della data di emissione del bando o dopo la sua scadenza. Di seguito si riporta l'implementazione:

```
CREATE OR REPLACE TRIGGER VERIFICA_PRENOTAZIONE_BORSA_STUDIO
BEFORE INSERT OR UPDATE ON partecipazione_bando_borsa FOR EACH ROW
DECLARE
    dataDomanda partecipazione_bando_borsa.data_domanda % TYPE := :NEW.data_domanda;
    tuplaBandoBorsa bando_borsa % ROWTYPE;
BEGIN
```

```
    --recupero la tupla di tale bando per cui lo studente intende partecipare
    SELECT * INTO tuplaBandoBorsa
    FROM bando_borsa
    WHERE bando_borsa.numero_bando_borsa = :NEW.numero_bando_borsa;

    IF (tuplaBandoBorsa.data_emissione > dataDomanda OR tuplaBandoBorsa.scadenza <
       dataDomanda) THEN
        RAISE NO_DATA_FOUND;
    END IF;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20020, 'Data prenotazione non valida');
END;
```

Verifica prenotazione bando Erasmus

Uno studente non può effetuare la domanda di partecipazione a un bando Erasmus prima della data di emissione del bando o dopo la sua scadenza. Di seguito si riporta l'implementazione:

```
CREATE OR REPLACE TRIGGER VERIFICA_PRENOTAZIONE_ERASMUS
BEFORE INSERT OR UPDATE ON partecipazione_bando_erasmus FOR EACH ROW
DECLARE
    dataDomanda partecipazione_bando_erasmus.data_domanda % TYPE := :NEW.data_domanda;
    tuplaBandoErasmus bando_erasmus % ROWTYPE;
BEGIN
    -- recupero la tupla di tale bando per cui lo studente intende partecipare
    SELECT * INTO tuplaBandoErasmus
    FROM bando_erasmus
    WHERE bando_erasmus.numero_bando_erasmus = :NEW.numero_bando_erasmus;

    IF (tuplaBandoErasmus.data_emissione > dataDomanda OR tuplaBandoErasmus.scadenza <
       dataDomanda) THEN
        RAISE NO_DATA_FOUND;
    END IF;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20030, 'Data prenotazione non valida');
END;
```

Verifica pagamento tassa ridondante

Uno studente non può pagare la stessa tassa due volte. Di seguito si riporta l'implementazione:

```
CREATE OR REPLACE TRIGGER VERIFICA_PAGAMENTO_TASSA
BEFORE UPDATE ON tassa FOR EACH ROW
DECLARE
    tuplaTassa tassa % ROWTYPE;
    taxAlreadyPaid EXCEPTION;
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
    -- recupero la tupla della tassa da pagare
    SELECT * INTO tuplaTassa
    FROM tassa
    WHERE :OLD.numero_fattura = tassa.numero_fattura;
    -- verifico che la tassa non sia già stata pagata
    IF ((tuplaTassa.IUV IS NOT NULL) AND (tuplaTassa.data_pagamento IS NOT NULL)) THEN
        RAISE taxAlreadyPaid;
    END IF;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20040, 'NO DATA FOUND');
    WHEN taxAlreadyPaid THEN
        RAISE_APPLICATION_ERROR(-20041, 'La tassa risulta essere pagata. Non devi
                                ripagarla!');
END;
```

Verifica compilazione questionario studente

Uno studente può compilare il questionario relativo a una determinata edizione di insegnamento insegnata da un docente se e solo se:

- ha frequentato l'edizione dell'insegnamento relativo
- l'edizione insegnamento e l'insegnamento siano relativi al corso di studi dello studente
- il docente per il quale si compila il questionario abbia effettivamente insegnato quell'edizione di insegnamento
- la data di compilazione è successiva alla data di inizio frequentazione del corso in questione
- lo studente non ha compilato ancora il questionario

Di seguito si riporta l'implementazione:

```
CREATE OR REPLACE TRIGGER trigger_questionario
BEFORE INSERT OR UPDATE ON questionario
FOR EACH ROW
DECLARE
      studentID                        questionario.matricola_studente % TYPE;
      studentCdL                       studente.codice_corso % TYPE;
      cod_ins                          offerta_insegnamento.codice_insegnamento % TYPE;
      tupla_FEI                        frequenta_edizione_insegnamento % ROWTYPE;
      tess_docente                     insegna_edizione.tesserino_docente % TYPE;
      data_inizio                      date;
      conteggioQuestionario            NUMBER(2, 0);

      invalidCdL                       EXCEPTION;
      invalid_FEI                      EXCEPTION;
      invalidProfessor                 EXCEPTION;
      invalidCompilationDate           EXCEPTION;
      alreadyCompiled                  EXCEPTION;
BEGIN
      -- controllare che l'edizione insegnamento/insegnamento sia relativo al corso di
      -- studi dello studente. Prelevo il CdL di tale studente
      SELECT codice_corso INTO studentCdL
      FROM studente
      WHERE matricola_studente = :NEW.matricola_studente;
      -- verifico che tale insegnamento appartenga a tale CdL
      SELECT codice_insegnamento INTO cod_ins
      FROM offerta_insegnamento
      WHERE codice_corso = studentCdL
      AND codice_insegnamento = :NEW.codice_insegnamento;

      IF (cod_ins IS NULL) THEN
            RAISE invalidCdL;
      END IF;

      --controllare che abbia frequentato l'edizione per lo stesso anno accademico
      SELECT matricola_Studente INTO studentID
      FROM frequenta_edizione_insegnamento
      WHERE codice_insegnamento = :NEW.codice_insegnamento AND
            anno_accademico = :NEW.anno_accademico AND
            matricola_studente = :NEW.matricola_studente;

      IF (studentID IS NULL) THEN
            RAISE invalid_FEI;
      END IF;

      -- controllare che il docente abbia insegnato quell'edizione
      SELECT tesserino_docente INTO tess_docente
      FROM insegna_edizione
      WHERE tesserino_docente = :NEW.tesserino_docente AND
            anno_accademico = :NEW.anno_accademico AND
            codice_insegnamento = :NEW.codice_insegnamento;
```

```
      IF (tess_docente IS NULL) THEN
            RAISE invalidProfessor;
      END IF;
      -- controllare che la data di compilazione sia successiva alla data di inizio
      -- frequentazione. Prelevo la data di inizio frequentazione di tale edizione
      -- insegnamento di tale studente
      SELECT data_ins INTO data_inizio
      FROM frequenta_edizione_insegnamento
      WHERE codice_insegnamento = :NEW.codice_insegnamento AND
            anno_accademico = :NEW.anno_accademico AND
            matricola_studente = :NEW.matricola_studente;

      IF :NEW.data_compilazione < data_inizio THEN
            RAISE invalidCompilationDate;
      END IF;

      -- controllare che non esista un questionario già compilato per tale edizione di
      -- insegnamento e per tale docente di tale studente
      SELECT cont INTO conteggioQuestionario
      FROM ((
               SELECT count(*) as cont FROM QUESTIONARIO
               WHERE tesserino_docente = :NEW.tesserino_docente AND
                      anno_accademico = :NEW.anno_accademico AND
                      codice_insegnamento = :NEW.codice_insegnamento AND
                      matricola_studente = :NEW.matricola_studente
               GROUP BY numero_questionario) UNION ALL SELECT 0 FROM DUAL)WHERE ROWNUM = 1;

       IF (conteggioQuestionario > 0) THEN
               RAISE alreadyCompiled;
       END IF;
EXCEPTION
      WHEN invalidCdL THEN
            RAISE_APPLICATION_ERROR(-20050, 'Insegnamento immesso non valido: non
                                    appartiene al CdL dello studente!');
      WHEN invalid_FEI THEN
            RAISE_APPLICATION_ERROR(-20051, 'Ed_Insegnamento immessa non valida: lo
                                     studente non ha ancora frequentato tale edizione!');
      WHEN invalidProfessor THEN
            RAISE_APPLICATION_ERROR(-20052, 'Docente immesso non valido: non ha insegnato
                                     tale edizione dell''insegnamento');
      WHEN invalidCompilationDate THEN
            RAISE_APPLICATION_ERROR(-20053, 'Data compilazione non valida: deve prima aver
                                    frequentato l''edizione dell''insegnamento');
      WHEN alreadyCompiled THEN
            RAISE_APPLICATION_ERROR(-20054, 'Hai già compilato il questionario!');
      WHEN NO_DATA_FOUND THEN
            RAISE_APPLICATION_ERROR(-20055, 'Non puoi compilare il questionario!');
END;
```

Verifica assegnazione bando Erasmus

Gli Erasmus sono assegnati agli studenti che hanno fatto richiesta per quel bando Erasmus. Inoltre, un bando Erasmus non deve essere assegnato ad uno studente che ha già vinto l'Erasmus in quell'anno accademico e il numero di bandi Erasmus assegnati non deve superare il numero di borse Erasmus disponibili (assegnabili). Di seguito si riporta l'implementazione:

```
CREATE OR REPLACE TRIGGER trigger_assegnazione_erasmus
BEFORE INSERT OR UPDATE ON assegnazione_erasmus FOR EACH ROW
DECLARE
    studentID               partecipazione_bando_erasmus.matricola_studente % TYPE;
    annoAccademico          date;
    fineAnnoAccademico      date;
    data_scadenza           date;
    nPosti                  bando_erasmus.numero_posti % TYPE;
```

```
    bando_assegnato         assegnazione_erasmus.numero_bando_erasmus % TYPE;
    nPostiAssegnati         bando_erasmus.numero_posti % TYPE := 0;

    invalidStudent          EXCEPTION;
    invalidDate             EXCEPTION;
    alreadyAssigned         EXCEPTION;
    exceed_nPosti           EXCEPTION;
BEGIN
    --recupero l'anno accademico in base alla data di assegnazione dell'erasmus
    IF (EXTRACT(MONTH FROM TO_DATE(:NEW.data_assegnazione, 'DD/MM/YYYY')) > 8) THEN
        SELECT TO_DATE(('01/09/' || SUBSTR(TO_CHAR(:NEW.data_assegnazione, 'DD/MM/YYYY'),
        4, 4)), 'DD/MM/YYYY') INTO annoAccademico FROM DUAL;

        SELECT ADD_MONTHS(TO_DATE(annoAccademico, 'DD/MM/YYYY'), 12) INTO
        fineAnnoAccademico FROM DUAL;
    ELSE
        SELECT TO_DATE(('01/09/' || SUBSTR(TO_CHAR(:NEW.data_assegnazione, 'DD/MM/YYYY'),
        4, 4)), 'DD/MM/YYYY') INTO fineAnnoAccademico FROM DUAL;
        SELECT ADD_MONTHS(TO_DATE(fineAnnoAccademico, 'DD/MM/YYYY'), -12) INTO
        annoAccademico FROM DUAL;
    END IF;

    --controllo che lo studente abbia fatto richiesta
    SELECT matricola_studente INTO studentID
    FROM partecipazione_bando_erasmus
    WHERE numero_bando_erasmus = :NEW.numero_bando_erasmus AND matricola_studente =
    :NEW.matricola_studente;


    IF (studentID IS NULL) THEN
        RAISE invalidStudent;
    END IF;

    --verifico che la data di assegnazione sia successiva alla scadenza
    -- prelevo la data di scadenza delle prenotazioni e il numero di posti per tale bando

    SELECT bando_erasmus.scadenza, bando_erasmus.numero_posti INTO data_scadenza, nPosti
    FROM bando_erasmus
    WHERE numero_bando_erasmus = :NEW.numero_bando_erasmus;

    IF data_scadenza >= :NEW.data_assegnazione THEN
        RAISE invalidDate;
    END IF;

   --controllo che lo studente non abbia ricevuto già un bando erasmus nell'anno accademico
    SELECT numero_bando_erasmus INTO bando_assegnato
    FROM assegnazione_erasmus
    WHERE(NOT EXISTS(
                      -- recupero eventuali eramus ricevuti nell'anno accademico
                    SELECT * FROM assegnazione_erasmus
                     WHERE matricola_studente = :NEW.matricola_studente
                     AND TO_DATE(data_assegnazione, 'DD/MM/YYYY') >=
                     TO_DATE(annoAccademico, 'DD/MM/YYYY')
                     AND TO_DATE(data_assegnazione, 'DD/MM/YYYY') <
                     TO_DATE(fineAnnoAccademico, 'DD/MM/YYYY')
     )) AND ROWNUM = 1;

    IF (bando_assegnato IS NULL) THEN
        RAISE alreadyAssigned;
    END IF;

    --verifico posti. Conto il numero di erasmus già assegnati
    SELECT COUNT(*) INTO nPostiAssegnati
    FROM assegnazione_erasmus
    WHERE numero_bando_erasmus = :NEW.numero_bando_erasmus;
```

```
        --verifico che ci siano ancora posti disponibili
    IF (nPostiAssegnati >= nPosti) THEN
        RAISE exceed_nPosti;
    END IF;

EXCEPTION
    WHEN invalidStudent THEN
        RAISE_APPLICATION_ERROR(-20060, 'Lo studente immesso non ha fatto richiesta per
                                tale bando!');
    WHEN invalidDate THEN
        RAISE_APPLICATION_ERROR(-20061, 'La data immessa non è valida per tale bando!');
    WHEN alreadyAssigned THEN
        RAISE_APPLICATION_ERROR(-20062, 'Per tale studente è già stata assegnato un
                                bando_erasmus nell''anno accademico corrente!');
    WHEN exceed_nPosti THEN
        RAISE_APPLICATION_ERROR(-20063, 'Numero massimo di posti assegnati per
                                bando_erasmus raggiunto!');
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20064, 'Non puoi assegnare il bando!');
END;
```

Verifica assegnazione bando borsa di studio

Le borse di studio sono assegnate agli studenti che hanno fatto richiesta per quel bando di borsa di studio. Inoltre, una borsa di studio non deve essere assegnata ad uno studente che ha già vinto la borsa di studio in quell'anno accademico e il numero di borse di studio assegnate non deve superare il numero di borse di studio disponibili (assegnabili). Di seguito si riporta l'implementazione:

```
CREATE OR REPLACE TRIGGER trigger_assegnazione_borse BEFORE INSERT OR UPDATE ON
assegnazione_borse
FOR EACH ROW
DECLARE
    studentID               partecipazione_bando_borsa.matricola_studente % TYPE;
    annoAccademico          date;
    fineAnnoAccademico      date;
    data_scadenza           date;
    nBorse                  bando_borsa.numero_borse % TYPE;
    borsa_assegnata         assegnazione_borse.numero_bando_borsa % TYPE;
    nBorseAssegnate         bando_borsa.numero_borse % TYPE := 0;

    invalidStudent          EXCEPTION;
    invalidDate             EXCEPTION;
    alreadyAssigned         EXCEPTION;
    exceed_nBorse           EXCEPTION;
BEGIN
    -- recupero l'anno accademico in base alla data di assegnazione della bds
    IF (EXTRACT(MONTH FROM TO_DATE(:NEW.data_assegnazione, 'DD/MM/YYYY')) > 8) THEN
        SELECT TO_DATE(('01/09/' || SUBSTR(TO_CHAR(:NEW.data_assegnazione, 'DD/MM/YYYY'),
        4, 4)), 'DD/MM/YYYY') INTO annoAccademico FROM DUAL;
        SELECT ADD_MONTHS(TO_DATE(annoAccademico, 'DD/MM/YYYY'), 12) INTO
        fineAnnoAccademico FROM DUAL;
    ELSE
        SELECT TO_DATE(('01/09/' || SUBSTR(TO_CHAR(:NEW.data_assegnazione, 'DD/MM/YYYY'),
        4, 4)), 'DD/MM/YYYY') INTO fineAnnoAccademico FROM DUAL;
        SELECT ADD_MONTHS(TO_DATE(fineAnnoAccademico, 'DD/MM/YYYY'), -12) INTO
        annoAccademico FROM DUAL;
    END IF;

    --controllo che lo studente abbia fatto richiesta
    SELECT matricola_studente INTO studentID
    FROM partecipazione_bando_borsa
    WHERE numero_bando_borsa = :NEW.numero_bando_borsa AND matricola_studente =
    :NEW.matricola_studente;
```

```
    IF (studentID IS NULL) THEN
        RAISE invalidStudent;
    END IF;

    --verifico che la data di assegnazione sia successiva alla scadenza. Prelevo la data di
    -- scadenza delle prenotazioni e il numero di borse assegnabili per tale bando
    SELECT bando_borsa.scadenza, bando_borsa.numero_borse INTO data_scadenza, nBorse
    FROM bando_borsa
    WHERE numero_bando_borsa = :NEW.numero_bando_borsa;

    -- la data di assegnazione deve essere successiva alla data di scadenza delle
    -- prenotazioni di tale bando
    IF data_scadenza >= :NEW.data_assegnazione THEN
        RAISE invalidDate;
    END IF;

    --controllo che lo studente non abbia ricevuto già una bds nell'anno accademico
    SELECT numero_bando_borsa INTO borsa_assegnata FROM assegnazione_borse
    WHERE (NOT EXISTS (-- recupero eventuali bds ricevute nell'anno accademico
                       SELECT *
                       FROM assegnazione_borse
                       WHERE matricola_studente = :NEW.matricola_studente
                       AND TO_DATE(data_assegnazione, 'DD/MM/YYYY') >=
                       TO_DATE(annoAccademico, 'DD/MM/YYYY')
                       AND TO_DATE(data_assegnazione, 'DD/MM/YYYY') <
                       TO_DATE(fineAnnoAccademico, 'DD/MM/YYYY')
     )) AND ROWNUM = 1;

    IF (borsa_assegnata IS NULL) THEN
        RAISE alreadyAssigned;
    END IF;

    --verifico posti. Conto il numero di borse già assegnate
    SELECT COUNT(*) INTO nBorseAssegnate FROM assegnazione_borse
    WHERE numero_bando_borsa = :NEW.numero_bando_borsa;

    --verifico che ci siano ancora borse assegnabili
    IF (nBorseAssegnate >= nBorse) THEN
        RAISE exceed_nBorse;
    END IF;

EXCEPTION
    WHEN invalidStudent THEN
        RAISE_APPLICATION_ERROR(-20070, 'Lo studente immesso non ha fatto richiesta per
                                tale bando!');
    WHEN invalidDate THEN
        RAISE_APPLICATION_ERROR(-20071, 'La data immessa non è valida per tale bando!');
    WHEN alreadyAssigned THEN
        RAISE_APPLICATION_ERROR(-20072, 'Per tale studente è già stata assegnata una borsa
                                di studio nell''anno accademico corrente!');
    WHEN exceed_nBorse THEN
        RAISE_APPLICATION_ERROR(-20073, 'Numero massimo di borse assegnate per bando_borsa
                                raggiunto!');
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20074, 'Non puoi assegnare il bando!');
END;
```

Verifica partecipazione seminario

Uno studente non può partecipare a più semminari conteporaneamente. Di seguito si riporta l'implementazione:

```
CREATE OR REPLACE TRIGGER VERIFICA_PARTECIPAZIONE_SEMINARIO
BEFORE INSERT OR UPDATE ON partecipa_seminario FOR EACH ROW
DECLARE
    numeroSeminariContemporanei NUMBER(4, 0);
    invalidParticipation EXCEPTION;
BEGIN
    -- VERIFICA CHE LO STUDENTE PARTECIPI GIA' AD UN ALTRO SEMINARIO NELLA STESSA DATA
    -- conto il numero dei seminari a cui già partecipa nella stessa data
    SELECT COUNT(*) INTO numeroSeminariContemporanei
    FROM partecipa_seminario
    WHERE partecipa_seminario.matricola_studente = :NEW.matricola_studente
    AND partecipa_seminario.data_seminario = :NEW.data_seminario;

    -- non deve partecipare a più seminari contemporaneamnte
    IF numeroSeminariContemporanei <> 0 THEN
        RAISE invalidParticipation;
    END IF;

EXCEPTION
     WHEN invalidParticipation THEN
        RAISE_APPLICATION_ERROR(-20080, 'Lo studente sta gia'' partecipando ad un
                                seminario');
END;
```

Si precisa che nella gestione delle eccezioni dei vari trigger appena visti e procedure che seguiranno, si è scelto di sfruttare i *raise application error.* I relativi codici di errore sono stati scelti in maniera opportuna riservando dieci codici di errore per ogni trigger. In maniera analoga verrà applicato lo stesso principio per le procedure. Pertanto il primo trigger ha a disposizione dieci codici di errore per la *raise application error.* Tale scelta è motivata dal fatto che, un'eventuale manutenzione di un trigger può far sì che vengano aggiunti un numero di ulteriori eccezioni più o meno consistente. Riservando alcuni codici di errore (che nel frattempo restano inutilizzati), si evita un'eventuale rimodulazione completa dei codici di errore su tutti i trigger e su tutte le procedure. Un'alternativa all'uso del *raise application error* consiste nell'usare delle semplici stampe a video contenenti dei messaggi diagnostici.

In particolare i codici di errori riservati per i trigger vanno da -20000 a -20099, mentre quelli per le procedure vanno da -20100 a -20199.

Procedure e funzioni

In questa sezione ci si occupa dell'automazione del database attraverso l'implementazione di procedure. Grazie all'implementazione di queste ultime, inserendo o cancellando una tupla in una singola tabella si riesce, se necessario, a modificare elementi anche nelle tabelle ad essa collegate. Nel caso in cui ci si occupi di cancellazione, di tuple o intere tabelle, va seguito l'ordine opposto rispetto a quello del popolamento. Il principale problema è stato capire come sfruttare al meglio il DB realizzato per trarne il massimo profitto.

Programmazione automatizzata appelli

Si tratta di una procedura che si occupa di andare ad automatizzare la programmazione degli appelli di una determinata edizione di uno specifico insegnamento dell'anno accademico corrente. Per fare ciò, si determinano delle date di appello d'esame tali per cui non vi è un altro appello d'esame di un'altra edizione insegnamento dello stesso anno di corso e dello stesso corso di laurea nella stessa data scelta. Si evitano anche i giorni considerati festivi dal punto di vista accademico: 'Sabato' e 'Domenica'. Una possibile estensione è quella di andare a evitare anche i giorni considerati festivi dal locale calendario regionale e/o nazionale.

In particolare, l'utente che avvia la procedura (docente, segreteria o amministratore) andrà a fissare il codice dell'insegnamento per il quale vuole programmare gli appelli, il numero massimo di studenti consentiti per ogni appello, il tipo d'appello (orale, scritto oppure non previsto) e il codice del corso di laurea in base al quale si vogliono evitare delle collisioni. Si noti che un insegnamento può appartenere a più corsi di laurea.

L'idea è quella di andare a determinare il mese attuale e l'anno accademico attuale in base alla data di sistema nel momento in cui viene lanciata la procedura, avviare un costrutto di ripetizione esterno che si arresterà quando il mese corrente di iterazione avrà raggiunto il mese di agosto (in cui si è supposto non possono essere fissati appelli). Per ogni mese si valuta giorno per giorno, se è possibile fissare l'appello in tale giorno di tale mese. Se non è possibile si "itera" sui giorni di tale mese. Chiaramente vi è un controllo per far sì che non vengano inseriti appelli "nel passato". Si noti che tale problematica vi è solo sul primo mese. Se non è possibile fissare l'appello in quel giorno di quel mese, si procede col giorno successivo e se il mese è terminato, si procede col mese successivo. Di seguito si riporta il codice implementativo:

```
CREATE OR REPLACE PROCEDURE procedura_programmazione_appelli (
    codice_insegnamento_input IN appello.codice_insegnamento % TYPE,
    max_studenti_input IN appello.max_studenti % TYPE,
    tipo_input IN appello.tipo % TYPE,
    codice_corso_input IN corso_laurea.codice_corso % TYPE) IS

    anno_accademico_input appello.anno_accademico % TYPE;
    giornoAttualeNumerico NUMBER(2, 0);
    giornoAttualeStringa CHAR(10);
    giornoFineMese NUMBER(2, 0);
    meseAttuale CHAR(15);
    monthCounter NUMBER(2, 0) := 0;
    annoAttuale CHAR(4);
    tempDate DATE;
    annoCorso edizione_insegnamento.anno_corso % TYPE;
    nAppelli NUMBER(5, 0);
    systemDate DATE;
    firstIteration NUMBER(1, 0) := 0;
BEGIN
    -- calcolo mese e anno accademico attuale da cui partire per la programmazione degli
    -- appelli
    systemDate := SYSDATE;
    SELECT TO_CHAR(systemDate, 'Month') INTO meseAttuale FROM DUAL;
```

```
SELECT LAST_DAY(TRUNC(systemDate, 'YYYY'), 5) INTO anno_accademico_input FROM DUAL;

-- prelievo anno corso dell'edizione insegnamento per cui si vuole avviare la procedura
-- di programmazione
SELECT edizione_insegnamento.anno_corso INTO annoCorso
FROM edizione_insegnamento JOIN offerta_insegnamento
ON edizione_insegnamento.codice_insegnamento =
   offerta_insegnamento.codice_insegnamento

WHERE edizione_insegnamento.anno_accademico = anno_accademico_input
   AND edizione_insegnamento.codice_insegnamento = codice_insegnamento_input
   AND offerta_insegnamento.codice_corso = codice_corso_input;

-- ciclo esterno che itera sui mesi
WHILE (meseAttuale != 'August')
LOOP
   -- calcolo l'ultimo giorno di tale mese
    SELECT TO_NUMBER(TO_CHAR(LAST_DAY(TO_DATE(LAST_DAY(systemDate, monthCounter),
           'DD/MM/YY')), 'DD')) INTO giornoFineMese FROM DUAL;

   -- alla prima iterazione, fissiamo la prima data utile per l'iterazione su tale
   -- mese come sysdate per evitare che vengano ipotizzate date di appello in giorni
   -- passati siccome l'iterazione parte dal primo giorno del mese

    IF (firstIteration = 1) THEN
        SELECT TO_NUMBER(TO_CHAR(TO_DATE(TRUNC(LAST_DAY(systemDate, monthCounter),
           'mm'), 'DD/MM/YY'), 'DD')) INTO giornoAttualeNumerico FROM DUAL;
    ELSE
        SELECT TO_NUMBER(TO_CHAR(systemDate, 'DD')) INTO giornoAttualeNumerico FROM
           DUAL;
        IF (giornoAttualeNumerico + 1 > giornoFineMese) THEN
            giornoAttualeNumerico := giornoFineMese + 1;
        ELSE
            giornoAttualeNumerico := giornoAttualeNumerico + 1;
        END IF;
        firstIteration := 1;
    END IF;
   -- ciclo interno che itera sui giorni del mese
    WHILE (giornoAttualeNumerico != giornoFineMese + 1)
    LOOP
    -- controllo che tale giorno non sia sabato o domenica
        SELECT TO_CHAR(TO_DATE(TRUNC(LAST_DAY(systemDate, monthCounter), 'mm') +
           giornoAttualeNumerico, 'DD/MM/YY'), 'Day') INTO giornoAttualeStringa FROM
           DUAL;
        IF (giornoAttualeStringa = 'Saturday' OR giornoAttualeStringa = 'Sunday') THEN
            giornoAttualeNumerico := giornoAttualeNumerico + 1;
            CONTINUE;
        END IF;
        -- recupero dell'anno
        SELECT TO_CHAR(EXTRACT (YEAR FROM TO_DATE(LAST_DAY(systemDate +
           giornoAttualeNumerico, monthCounter), 'DD/MM/YY'))) INTO annoAttuale FROM
           DUAL;
        -- costruizione della data ipotetica
        tempDate := TO_DATE(TO_CHAR(giornoAttualeNumerico || meseAttuale ||
           annoAttuale), 'DD/MM/YYYY');

        -- automatizzazione appelli
        SELECT conteggioAppelli INTO nAppelli FROM (
           SELECT conteggioAppelli FROM (
                  -- conto il numero di appelli già fissati nella data ipotetica
                  -- per edizioni di insegnamento del medesimo corso di laurea ed anno
                  -- di corso
                  SELECT COUNT(*) as conteggioAppelli FROM appello JOIN
                  edizione_insegnamento ON appello.anno_accademico =
                  edizione_insegnamento.anno_accademico AND appello.codice_insegnamento
                  = edizione_insegnamento.codice_insegnamento
```

```
                    JOIN offerta_insegnamento ON offerta_insegnamento.codice_insegnamento
                    = edizione_insegnamento.codice_insegnamento

                    WHERE appello.anno_accademico = anno_accademico_input
                    AND appello.data_appello = tempDate
                    AND offerta_insegnamento.codice_corso = codice_corso_input
                    AND edizione_insegnamento.anno_corso = annoCorso
                    GROUP BY appello.data_appello)
                UNION ALL SELECT 0 FROM DUAL)
            WHERE ROWNUM = 1;

/* se esistono già appelli con suddette caratteristiche, si passa al giorno successivo
e vengono ripetuti i controlli su tale giorno, altrimenti è possibile fissare tale appello
in tale giorno */
            IF (nAppelli > 0) THEN
                giornoAttualeNumerico := giornoAttualeNumerico + 1;
                CONTINUE;
            END IF;
            -- inserimento
            INSERT INTO APPELLO(ANNO_ACCADEMICO, DATA_APPELLO, CODICE_INSEGNAMENTO,
              DATA_INIZIO, DATA_FINE, MAX_STUDENTI, TIPO) VALUES
            (anno_accademico_input, tempDate, codice_insegnamento_input, tempDate - 30,
              tempDate - 5, max_studenti_input, tipo_input);
            EXIT;
        END LOOP;
/* incremento del mese sia nel caso in cui non sia possibile fissare l'appello in tale mese
poichè non vi sono date disponibili oppure nel caso in cui sia stata fissata la data
correttamente */
        --riposizionamento al primo giorno del mese
        giornoAttualeNumerico := 1;
        monthCounter := monthCounter + 1;
        SELECT TO_CHAR(TO_DATE(LAST_DAY(systemDate, monthCounter), 'DD/MM/YY'), 'Month')
              INTO meseAttuale FROM DUAL;
    END LOOP;
    COMMIT;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        raise_application_error(-20110, 'Non esiste l''edizione insegnamento o
                                l''insegnamento.');
        ROLLBACK;
    WHEN OTHERS THEN
        raise_application_error(-20111, 'Non è stato possibile stabilire tutte le date di
                                appello per l''anno accademico.');
        ROLLBACK;
END;
```

Assegnazione docente edizione insegnamento

La procedura consente di assegnare il docente per una specifica edizione di un insegnamento. Se non esistono questionari coompilati di tale edizione insegnamento (in tal caso si presuppone che tale insegnamento sia nuovo), allora viene scelto il docente migliore tra i docenti degli altri insegnamenti principali, relativi allo stesso corso di laurea dell'insegnamento fornito in input. Se esistono dei questionari compilati di tale insegnamento (edizioni insegnamento passate), si assegna il docente che ha la migliore valutazione generale tra i questionari relativi a tale insegnamento, cioè il docente che tra tutti i docenti delle edizioni di insegnamento passate ha avuto la valutazione migliore. Si presuppone che l'anno accademico venga espresso nello stesso formato che si usa nel DML, cioè `TO_DATE('YYYY')`. Inoltre in entrambe le casistiche, non è detto che il docente migliore venga assegnato. Infatti è possibile che insegni abbastanza edizioni insegnamento e quindi va scelto il secondo miglior docente tra quelli possibili. In particolare, si presuppone che se il docente selezionato insegni al più cinque insegnamenti, allora non può essere preso in considerazione per essere assegnato a questa nuova edizione insegnamento.

Il docente migliore in entrambi i casi è stabilito sulla base della media delle somme delle medie degli indici di "gradimento" sui quali è basato il questionario somministrato agli studenti. Cioè si effettua la media delle somme dei valori di un singolo indice (i valori degli indici appartengono a [1, 5]), dopodiché si effettua la media delle medie dei valori degli indici di gradimento. Ordinando in senso decrescente per media di medie la relazione risultante, si ottiene che la prima riga conterrà il miglior docente da selezionare. Qualora non sia possibile farlo, si passa alla riga successiva finché non si è trovato il docente da assegnare oppure non sono stati esauriti i docenti da prendere in considerazione.

```
CREATE OR REPLACE PROCEDURE procedura_assegnazione_docente_insegna_edizione (
        codice_insegnamento_input IN questionario.codice_insegnamento % TYPE,

        tipo_docente_input IN insegna_edizione.tipo_docente % TYPE,
        anno_accademico_input IN insegna_edizione.anno_accademico % TYPE,

        semestre_input IN edizione_insegnamento.semestre % TYPE,
        anno_corso_input IN edizione_insegnamento.anno_corso % TYPE,
        CFU_input IN edizione_insegnamento.CFU % TYPE,
        svolgimento_input IN edizione_insegnamento.svolgimento % TYPE
    ) IS

    tesserino_docente_output            questionario.tesserino_docente % TYPE;
    annoAccademico                      date;
    checkValue                          questionario.tesserino_docente % TYPE;
    numeroQuestionariTrovati            NUMBER(5, 0);
    nDocentiEdizione                    NUMBER(5, 0);
    nDocentiCdL                         NUMBER(5, 0);
    docenteTrovato                      CHAR(1) := 'F';
    i                                   NUMBER(5, 0) := 1;
    insegnamentiInsegnati               NUMBER(5, 0) := 0;
    docenteNonDisponibile               EXCEPTION;
BEGIN
--inserimento edizione insegnamento nell'anno accademico di input
    annoAccademico := TO_DATE(anno_accademico_input, 'DD/MM/YYYY');
    INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
       ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
    (codice_insegnamento_input, TO_DATE(annoAccademico, 'DD/MM/YYYY'), semestre_input,
       anno_corso_input, CFU_input, svolgimento_input);
    -- conteggio questionari
    SELECT conteggio INTO numeroQuestionariTrovati
    FROM ((     -- conto il numero di questionari per tale insegnamento
              SELECT COUNT(*) AS conteggio
                FROM questionario
                WHERE questionario.codice_insegnamento = codice_insegnamento_input
                GROUP BY questionario.codice_insegnamento
        ) UNION ALL SELECT 0 FROM DUAL
    ) WHERE ROWNUM = 1;

    -- conteggio docenti edizione
    SELECT conteggio INTO nDocentiEdizione FROM
        (
       -- conto il numero di docenti per i quali sono stati compilati dei questionari in
       -- tale insegnamento
         SELECT COUNT(*) AS conteggio FROM
            ((  -- recupero i docenti per i quali sono stati compilati dei questionari in
                 -- tale insegnamento
                SELECT questionario.tesserino_docente
                FROM questionario
                WHERE codice_insegnamento = codice_insegnamento_input
                GROUP BY questionario.tesserino_docente
            )) UNION ALL SELECT 0 FROM DUAL)
    WHERE ROWNUM = 1;
```

```
    -- conteggio docenti CdL possibili. Conto i docenti, che rispettano le condizioni, di
    -- tale CdL

    SELECT conteggio INTO nDocentiCdL FROM
        (SELECT COUNT(*) AS conteggio FROM
            (( -- recupero i docenti di uno dei CdL a cui tale insegnamento appartiene che
              -- insegnano materie principali in tale CdL

              SELECT questionario.tesserino_docente
               FROM offerta_insegnamento JOIN questionario
                       ON offerta_insegnamento.codice_insegnamento =
                            questionario.codice_insegnamento
                       WHERE offerta_insegnamento.codice_corso = (
                           SELECT offerta_insegnamento.codice_corso
                           FROM offerta_insegnamento
                           WHERE offerta_insegnamento.codice_insegnamento =
                            codice_insegnamento_input AND ROWNUM = 1)
                       AND offerta_insegnamento.corso_principale = 'Y'
                GROUP BY questionario.tesserino_docente
            )) UNION ALL SELECT 0 FROM DUAL)
    WHERE ROWNUM = 1;

    -- presenza questionari. Verifico se ci sono questionari per tale insegnamento
    IF (numeroQuestionariTrovati > 0) THEN
/* non è un nuovo insegnamento: si ricerca nelle edizioni insegnamento passate il miglior
docente. Per tale motivo viene stabilito un indice di valutazione complessivo del docente.
Tale indice è la media delle medie delle valutazioni ottenute in tale insegnamento
attraverso i questionari */
        WHILE (docenteTrovato = 'F')
        LOOP
/* recupero la lista dei docenti migliori secondo tale indice. Parto dal primo docente e
verifico se può effettivamente insegnare una nuova edizione di insegnamento */
            SELECT tesserinoDocenteSelezionato INTO tesserino_docente_output FROM
                (SELECT tesserinoDocenteSelezionato, ROWNUM AS RN FROM (
                    SELECT questionario.tesserino_docente AS tesserinoDocenteSelezionato,
                      (((SUM(questionario.gradimento) / COUNT(questionario.gradimento)) +
                        (SUM(questionario.disponibilita_docente) /
                             COUNT(questionario.disponibilita_docente)) +
                        (SUM(questionario.precisione_orario) /
                             COUNT(questionario.precisione_orario)) +
                        (SUM(questionario.materiale_didattico) /
                             COUNT(questionario.materiale_didattico))) / 4) AS Media

                        FROM questionario
                        WHERE codice_insegnamento = codice_insegnamento_input
                        GROUP BY questionario.tesserino_docente
                        ORDER BY Media DESC
            )) WHERE RN = i;

            -- conto quante cattedre ha già tale docente nell'anno accademico
            SELECT conteggioInsegnamentiDocente INTO insegnamentiInsegnati
            FROM
                ((  SELECT COUNT(*) AS conteggioInsegnamentiDocente
                    FROM docente JOIN insegna_edizione
                    ON docente.numero_tesserino = insegna_edizione.tesserino_docente
                    JOIN edizione_insegnamento
                    ON edizione_insegnamento.anno_accademico =
                      insegna_edizione.anno_accademico
                    AND edizione_insegnamento.codice_insegnamento =
                      insegna_edizione.codice_insegnamento
                    WHERE edizione_insegnamento.anno_accademico = anno_accademico_input
                    AND docente.numero_tesserino = tesserino_docente_output
                    GROUP BY insegna_edizione.tesserino_docente) UNION ALL SELECT 0 FROM
```

```
                    DUAL)
            WHERE ROWNUM = 1;
/* se tale docente non ha raggiunto il numero massimo di cattedre (5), allora è possibile
assegnarlo come docente. Altrimenti non assegna tale docente e passa al prossimo docente
valido per tale insegnamento */
            IF (insegnamentiInsegnati > 4 AND (nDocentiEdizione - i) < 1) THEN
                RAISE docenteNonDisponibile;
            ELSIF (insegnamentiInsegnati <= 4) THEN
                docenteTrovato := 'T';
                INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
                      TESSERINO_DOCENTE, TIPO_DOCENTE) VALUES
                (codice_insegnamento_input, TO_DATE(annoAccademico, 'DD/MM/YYYY'),
                      tesserino_docente_output, tipo_docente_input);
            ELSIF (nDocentiCdL - i) < 1) THEN
                RAISE docenteNonDisponibile;
            ELSE
                i := i + 1;
            END IF;
        END LOOP;
    ELSE

/* assenza questionari. Se non ci sono questionari per tale insegnamento, è un nuovo
insegnamento. Il docente NON va ricercato nelle vecchie edizioni insegnamento. Per tale
motivo viene stabilito un indice di valutazione complessivo del docente. Tale indice è la
media delle medie delle valutazioni ottenute attraverso i questionari in un insegnamento
principale di uno dei CdL a cui appartiene tale insegnamento */
        WHILE (docenteTrovato = 'F')
        LOOP
            -- recupero la lista dei docenti migliori secondo tale indice
            -- parto dal primo docente e verifico se può effettivamente insegnare una nuova
            -- edizione di insegnamento

            SELECT tesserinoDocenteSelezionato INTO tesserino_docente_output FROM
                (SELECT tesserinoDocenteSelezionato, ROWNUM AS RN FROM (
                    SELECT questionario.tesserino_docente AS tesserinoDocenteSelezionato,
                      (((SUM(questionario.gradimento) / COUNT(questionario.gradimento)) +
                        (SUM(questionario.disponibilita_docente) /
                             COUNT(questionario.disponibilita_docente)) +
                        (SUM(questionario.precisione_orario) /
                             COUNT(questionario.precisione_orario)) +
                        (SUM(questionario.materiale_didattico) /
                             COUNT(questionario.materiale_didattico))) / 4) AS Media

                        FROM offerta_insegnamento JOIN questionario
                        ON offerta_insegnamento.codice_insegnamento =
                             questionario.codice_insegnamento
                        WHERE offerta_insegnamento.codice_corso = (
                            SELECT offerta_insegnamento.codice_corso
                            FROM offerta_insegnamento
                            WHERE offerta_insegnamento.codice_insegnamento =
                             codice_insegnamento_input AND ROWNUM = 1)
                        AND offerta_insegnamento.corso_principale = 'Y'
                        GROUP BY questionario.tesserino_docente
                        ORDER BY Media DESC
            )) WHERE RN = i;
            -- conto quante cattedre ha già tale docente nell'anno accademico
            SELECT conteggioInsegnamentiDocente INTO insegnamentiInsegnati
            FROM
                ((  SELECT COUNT(*) AS conteggioInsegnamentiDocente
                    FROM docente JOIN insegna_edizione
                    ON docente.numero_tesserino = insegna_edizione.tesserino_docente
                    JOIN edizione_insegnamento
                    ON edizione_insegnamento.anno_accademico =
                      insegna_edizione.anno_accademico
                    AND edizione_insegnamento.codice_insegnamento =
                      insegna_edizione.codice_insegnamento
```

```
                    WHERE edizione_insegnamento.anno_accademico = anno_accademico_input
                    AND docente.numero_tesserino = tesserino_docente_output
                    GROUP BY insegna_edizione.tesserino_docente) UNION ALL SELECT 0 FROM
                DUAL)
            WHERE ROWNUM = 1;

/* se tale docente non ha raggiunto il numero massimo di cattedre (5), allora è possibile
assegnarlo come docente. Altrimenti non assegna tale docente e passa al prossimo docente
valido tra i possibili docenti del CdL estratti precedentemente */

            IF (insegnamentiInsegnati > 4 AND (nDocentiCdL - i) < 1) THEN
                RAISE docenteNonDisponibile;
            ELSIF (insegnamentiInsegnati <= 4) THEN
                docenteTrovato := 'T';
                INSERT INTO INSEGNA_EDIZIONE(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO,
                      TESSERINO_DOCENTE, TIPO_DOCENTE) VALUES
                (codice_insegnamento_input, TO_DATE(annoAccademico, 'DD/MM/YYYY'),
                      tesserino_docente_output, tipo_docente_input);
            ELSIF (nDocentiCdL - i) < 1) THEN
                RAISE docenteNonDisponibile;
            ELSE
                i := i + 1;
            END IF;
        END LOOP;
    END IF;
    COMMIT;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        raise_application_error(-20180, 'NO DATA FOUND');
        ROLLBACK;
    WHEN docenteNonDisponibile THEN
        raise_application_error(-20181, 'Non esiste docente che può insegnare questa
edizione di insegnamento e che ha insegnato le precedenti edizioni insegnamento!');
        ROLLBACK;
    WHEN OTHERS THEN
        raise_application_error(-20182, 'Non è stato possibile assegnare un docente a tale
edizione di tale insegnamento.');
        ROLLBACK;
END;
```

Inserimento di un insegnamento

Procedura che si occupa di creare un insegnamento e una sua edizione insegnamento in termini di tuple all'interno delle rispettive tabelle. Inoltre permette di associare tale insegnamento a un particolare corso di laurea esistente con la possibilità di specificare se è caratterizzante o meno per tale CdL.

```
CREATE OR REPLACE PROCEDURE procedura_inserimento_insegnamento (
    codice_insegnamento_input IN insegnamento.codice_insegnamento % TYPE,
    nome_insegnamento_input IN insegnamento.nome % TYPE,
    anno_accademico_input IN edizione_insegnamento.anno_accademico % TYPE,
    semestre_input IN edizione_insegnamento.semestre % TYPE,
    anno_corso_input IN edizione_insegnamento.anno_corso % TYPE,
    CFU_input IN edizione_insegnamento.CFU % TYPE,
    svolgimento_input IN edizione_insegnamento.svolgimento % TYPE,
    codice_corso_input IN corso_laurea.codice_corso % TYPE,
    corso_principale_input IN offerta_insegnamento.corso_principale % TYPE
    ) IS
BEGIN
    INSERT INTO INSEGNAMENTO(CODICE_INSEGNAMENTO, NOME) VALUES  (codice_insegnamento_input,
       nome_insegnamento_input);
    INSERT INTO EDIZIONE_INSEGNAMENTO(CODICE_INSEGNAMENTO, ANNO_ACCADEMICO, SEMESTRE,
       ANNO_CORSO, CFU, SVOLGIMENTO) VALUES
    (codice_insegnamento_input, TO_DATE(anno_accademico_input, 'YYYY'), semestre_input,
       anno_corso_input, CFU_input, svolgimento_input);
```

```
    INSERT INTO OFFERTA_INSEGNAMENTO(CODICE_INSEGNAMENTO, CODICE_CORSO, CORSO_PRINCIPALE)
       VALUES (codice_insegnamento_input, codice_corso_input, corso_principale_input);
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN
        raise_application_error(-20120, 'Non è stato possibile inserire l''insegnamento,
                                l''edizione insegnamento specificata o l''associazione con
                                il CdL specificato.');
       ROLLBACK;
END;
```

Assegnazione di un tutor interno per un tirocinio interno

Uno studente che richiede di effettuare un tirocnio interno ha bisogno di un docente come tutor. Si suppone che venga assegnato allo studente un docente come tutor di tirocinio interno, che fra tutti i docenti è quello che ha fatto meno volte il tutor di tirocini interni. Si suppone che sia presente la seguente regola di business: uno studente non può richiedere il tirocinio se non ha conseguito almeno 120 CFU se la laurea è triennale oppure 80 CFU se si tratta di una laurea magistrale.

```
CREATE OR REPLACE PROCEDURE procedura_assegnazione_tutor_docente_tirocinio_interno
  (matricola_studente_input IN studente.matricola_studente % TYPE)
IS
   CFUconseguiti NUMBER(3, 0);
   tipoLaurea corso_laurea.tipo % TYPE;
   tesserino_docente_selezionato docente.numero_tesserino % TYPE;
BEGIN
/* verifica relativa ai CFU da conseguire. Calcolo la somma dei CFU ottenuti dallo studente
e nel caso in cui sia inferiore a 120 per uno studente della triennale o inferiore a 80 per
uno studente della magistrale, allora annullo l'inserimento */

    -- calcolo dei CFU conseguiti
    SELECT NVL(sommaCFU, 0) INTO CFUconseguiti
    FROM (
       -- recupero i CFU delle edizioni insegnamento che tale studente ha conseguito
       -- superando tali esami
       SELECT studente.matricola_studente, NVL(SUM(edizione_insegnamento.CFU), 0) AS
              sommaCFU
       FROM esame_superato JOIN edizione_insegnamento ON
       edizione_insegnamento.anno_accademico = esame_superato.anno_accademico
       AND edizione_insegnamento.codice_insegnamento = esame_superato.codice_insegnamento
       JOIN insegnamento ON esame_superato.codice_insegnamento =
       insegnamento.codice_insegnamento RIGHT JOIN studente
       ON studente.matricola_studente = esame_superato.matricola_studente
       WHERE studente.matricola_studente = matricola_studente_input
       GROUP BY studente.matricola_studente
    );
    -- verifico se lo studente è iscritto a un CdL triennale o magistrale e se ha
    -- abbastanza CFU
    SELECT corso_laurea.tipo INTO tipoLaurea FROM corso_laurea JOIN studente
    ON corso_laurea.codice_corso = studente.codice_corso
    WHERE studente.matricola_studente = matricola_studente_input;
    -- verifico tipo di laurea dello studente
    IF (LOWER(tipoLaurea) = TO_CHAR('triennale')) THEN
        IF (CFUconseguiti < 120) THEN
            ROLLBACK;
        END IF;
    ELSE
        IF (CFUconseguiti < 80) THEN
            ROLLBACK;
        END IF;
    END IF;
```

```
    -- prelievo docente che ha meno tirocini associati
    SELECT tesserino_docente INTO tesserino_docente_selezionato
    FROM (  SELECT docente.numero_tesserino AS tesserino_docente, COUNT(*) as
              conteggioTirocini
            FROM docente JOIN tirocinio
            ON docente.numero_tesserino = tirocinio.tesserino_docente
            GROUP BY docente.numero_tesserino
            ORDER BY conteggioTirocini
        )   WHERE ROWNUM = 1;

    -- si inserisce una tupla in tirocinio. Si assume che la durata media è di 3 mesi.
    INSERT INTO TIROCINIO(NUMERO_TIROCINIO, CFU, DATA_INIZIO, DATA_FINE, TESSERINO_DOCENTE,
       TESSERINO_TUTOR_AZIENDA, MATRICOLA_STUDENTE) VALUES
    (matricola_studente_input, 12, TO_DATE(NEXT_DAY(systemDate, 'MONDAY'), 'DD/MM/YYYY'),
       TO_DATE(systemDate + 90,'DD/MM/YYYY'), TO_CHAR(tesserino_docente_selezionato),NULL,
       TO_CHAR(matricola_studente_input));

    COMMIT;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        raise_application_error(-20160, 'Non è stato possibile determinare il docente da
              assegnare come tutor del tirocinio a questo studente.');
        ROLLBACK;
END;
```

Assegnazione di un tutor interno e un tutor aziendale (esterno) per un tirocinio esterno

Uno studente che richiede di effettuare un tirocnio esterno ha bisogno di un docente come tutor interno ed un tutor aziendale esterno. Si suppone che venga assegnato allo studente un docente come tutor di tirocinio interno, che fra tutti i docenti è quello che ha fatto meno volte il tutor di tirocini interni. Lo stesso schema viene adottato anche per l'assegnazione del tutor aziendale.

```
CREATE OR REPLACE PROCEDURE procedura_assegnazione_tutor_tirocinio (
    matricola_studente_input IN studente.matricola_studente % TYPE,
    partitaIva_input IN tutor_Aziendale.partita_iva % TYPE) IS

    tesserinoTutor_selezionato docente.numero_tesserino % TYPE;
    numeroTirocinio tirocinio.numero_tirocinio % TYPE;
BEGIN
    -- prelievo tutor Aziendale che ha meno tirocini associati dell'AZIENDA richiesta
    SELECT numeroTesserino INTO tesserinoTutor_selezionato
    FROM (
            SELECT tutor_Aziendale.partita_iva, tutor_Aziendale.numero_tesserino AS
              numeroTesserino, COUNT(*) AS conteggioTirocini
            FROM tutor_Aziendale JOIN tirocinio ON tutor_Aziendale.numero_tesserino =
              tirocinio.tesserino_tutor_AZIENDA
            WHERE tutor_Aziendale.partita_iva = partitaIva_input
            GROUP BY tutor_Aziendale.numero_tesserino, tutor_Aziendale.partita_iva
            ORDER BY COUNT(tutor_Aziendale.numero_tesserino)
        )
    WHERE ROWNUM = 1;

    --richiamo la procedura per l'assegnazione del tutor interno
    procedura_assegnazione_tutor_docente_tirocinio_interno(matricola_studente_input);

    --aggiorno la tupla appena creata nel tirocinio aggiungendo il tutor Aziendale
    -- precedentemente trovato. Si assume che la durata media sia 3 mesi
    SELECT numero_tirocinio INTO numeroTirocinio
    FROM tirocinio
    WHERE matricola_studente = matricola_studente_input;

    UPDATE tirocinio
```

```
    SET tesserino_tutor_AZIENDA = tesserinoTutor_selezionato
    WHERE numero_tirocinio = numeroTirocinio;

    COMMIT;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        raise_application_error(-20140, 'Non è stato possibile determinare il tutor
                                Aziendale da assegnare come tutor del tirocinio a questo
                                studente.');
        ROLLBACK;
END;
```

Calcolo statistiche studente

Procedura che calcola le statistiche degli esami e della laurea per uno studente la cui matricola viene passata in input, le stampa e le restituisce come parametri di output al chiamante.

```
CREATE OR REPLACE PROCEDURE procedura_calcola_statistiche_esami_e_laurea
(matricola_studente_input IN studente.matricola_studente % TYPE,
 mediaAritmeticaEsami OUT NUMBER, mediaPonderataEsami OUT NUMBER,
 mediaAritmeticaLaurea OUT NUMBER, mediaPonderataLaurea OUT NUMBER,
 progressioneCFU OUT NUMBER) IS

    esamiRegistrati NUMBER(2, 0);
    CFUconseguiti NUMBER(3, 0);
    sommaVotiEsami NUMBER(4, 0);
    combinazioneLineareVotiPesi NUMBER(5, 0);
    tipoLaurea corso_laurea.tipo % TYPE;
    divisoreProgressioneCFU NUMBER(3, 0);
BEGIN
    -- memorizzazione informazioni relative alla carriera di tale studente
    SELECT sommaCFU, conteggioEsami, sommaVoti, combLineare INTO CFUconseguiti,
       esamiRegistrati, sommaVotiEsami, combinazioneLineareVotiPesi
    FROM (
/* recupero, a partire dagli esami superati dagli studenti, le informazioni relative a tali
esami e alle edizioni di insegnamento associate e prelevo tali informazioni per lo studente
specifico, ovvero la matricola di input. In particolare, recupero:
    -   il numero di CFU conseguiti relativi agli insegnamenti del CdL a cui è iscritto
        tale studente
    -   il numero di esami registrati con successo
    -   la somma dei voti ottenuti a tali esami
    -   la combinazione lineare dei voti ottenuti e i CFU conseguiti per tali esami */
            SELECT studente.matricola_studente, SUM(edizione_insegnamento.CFU) AS sommaCFU,
               COUNT(*) AS conteggioEsami, SUM(esame_superato.voto) AS sommaVoti,
               SUM(esame_superato.voto * edizione_insegnamento.CFU) AS combLineare
             FROM esame_superato JOIN edizione_insegnamento
             ON edizione_insegnamento.anno_accademico = esame_superato.anno_accademico
                 AND edizione_insegnamento.codice_insegnamento =
                       esame_superato.codice_insegnamento
             JOIN insegnamento
             ON esame_superato.codice_insegnamento = insegnamento.codice_insegnamento
             JOIN studente
             ON studente.matricola_studente = esame_superato.matricola_studente
             WHERE esame_superato.matricola_studente = matricola_studente_input
             GROUP BY studente.matricola_studente
    );

    -- verifico se lo studente è iscritto a un CdL triennale o magistrale
    SELECT corso_laurea.tipo INTO tipoLaurea
    FROM corso_laurea JOIN studente
    ON corso_laurea.codice_corso = studente.codice_corso
    WHERE studente.matricola_studente = matricola_studente_input;
```

```
    IF (LOWER(tipoLaurea) = 'triennale') THEN
        divisoreProgressioneCFU := 180;
    ELSE
        divisoreProgressioneCFU := 120;
    END IF;

    -- calcolo statistiche della carriera di tale studente attraverso le informazioni
    -- ottenute in precedenza
    mediaAritmeticaEsami := ROUND(sommaVotiEsami / esamiRegistrati);
    mediaPonderataEsami := ROUND(combinazioneLineareVotiPesi / CFUconseguiti);
    mediaAritmeticaLaurea := ROUND((mediaAritmeticaEsami * 110) / 30);
    mediaPonderataLaurea := ROUND((mediaPonderataEsami * 110) / 30);
    progressioneCFU := TRUNC(((100 * CFUconseguiti) / divisoreProgressioneCFU), 2);

    -- visualizzazione statistiche
    DBMS_OUTPUT.PUT_LINE('Esami Registrati: ' || esamiRegistrati);
    DBMS_OUTPUT.PUT_LINE('CFU Conseguiti: ' || CFUconseguiti);
    DBMS_OUTPUT.PUT_LINE('Media Aritmetica Esami: ' || mediaAritmeticaEsami);
    DBMS_OUTPUT.PUT_LINE('Media Ponderata Esami: ' || mediaPonderataEsami);
    DBMS_OUTPUT.PUT_LINE('Media Aritmetica Laurea: ' || mediaAritmeticaLaurea);
    DBMS_OUTPUT.PUT_LINE('Media Ponderata Laurea: ' || mediaPonderataLaurea);
    DBMS_OUTPUT.PUT_LINE('Progressione CFU: ' || progressioneCFU || '%');
    COMMIT;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        raise_application_error(-20130, 'Non è possibile visualizzare le statistiche dei
                                tuoi esami e relative alla tua futura laurea');
        ROLLBACK;
END;
```

Viste

Le viste sono tabelle virtuali utili a regolamentare l'accesso ed aumentare la fruibilità dei dati per i vari utenti. Si possono usare viste per mostrare solo una porzione dei dati presenti in una tabella, per applicare funzioni più o meno complesse, per elaborare attributi derivati. Nel caso del DB *esse4,* le viste implementate sono le seguenti:

Vista Bandi Borsa di Studio Attivi

La prima vista implementata è quella che permette a uno studente di visualizzare i bandi di borsa di studio attualmente in corso (cioè che sono stati emessi e non sono ancora scaduti).

```
-- Vista dei Bandi di Borsa di Studio attivi
CREATE OR REPLACE VIEW vista_bando_borsa_studio AS
  SELECT *
  FROM bando_borsa
  WHERE TRUNC(data_emissione) < TRUNC(SYSDATE) AND TRUNC(scadenza) > TRUNC(SYSDATE);
```

Vista Bandi Erasmus Attivi

La seconda vista implementata è quella che permette a uno studente di visualizzare i bandi Erasmus attualmente in corso (cioè che sono stati emessi e non sono ancora scaduti).

```
-- Vista dei Bandi Erasmus attivi
CREATE OR REPLACE VIEW vista_bando_erasmus AS
  SELECT *
  FROM bando_erasmus
  WHERE TRUNC(data_emissione) < TRUNC(SYSDATE) AND TRUNC(scadenza) > TRUNC(SYSDATE);
```

Seminari del prossimo mese

La terza vista che è stata implementata è quella che permette di verificare quali seminari si svolgeranno nel prossimo mese, in maniera tale da organizzarsi con sufficiente anticipo.

```
-- Vista dei Seminari che avranno luogo nel prossimo mese
CREATE OR REPLACE VIEW vista_seminari_prossimo_mese AS
  SELECT data_seminario, nome, CFU, max_persone
  FROM seminario
  WHERE TRUNC(data_seminario) >= TRUNC(SYSDATE) AND
        TRUNC(data_seminario) < TRUNC(SYSDATE) + 30;
```

Corsi di Laurea con posti disponibili

Un Corso di Laurea con posti disponibili è un Corso di Laurea al quale si sono iscritti un numero di studenti inferiore rispetto alla capienza massima prevista. Questa vista permette a uno studente che pensa di iscriversi a un nuovo o differente Corso di Laurea, oppure a un docente di verificare quali Corsi di Laurea hanno posti ancora disponibili. Tale vista può essere usata anche da un utente esterno (magari uno studente non ancora iscritto. Si noti che non si considera quest'ultimo caso).

```
-- Vista dei Corsi di Laurea con posti disponibili
CREATE OR REPLACE VIEW vista_corsi_di_laurea_posti_disponibili AS
  SELECT corso_laurea.codice_corso, corso_laurea.tipo, corso_laurea.capienza,
         corso_laurea.nome, COUNT(*) AS studenti_iscritti
  FROM studente JOIN corso_laurea ON studente.codice_corso = corso_laurea.codice_corso
  GROUP BY corso_laurea.codice_corso, corso_laurea.tipo, corso_laurea.capienza,
           corso_laurea.nome;
```

Data Control Language

Si allegano i privilegi assegnati allo studente:

```
GRANT CONNECT, CREATE SESSION TO c##studente;
GRANT SELECT ON corso_laurea TO c##studente;
GRANT SELECT ON docente TO c##studente;
GRANT SELECT ON insegnamento TO c##studente;
GRANT SELECT ON edizione_insegnamento TO c##studente;
GRANT SELECT ON offerta_insegnamento TO c##studente;
GRANT SELECT ON insegna_edizione TO c##studente;
GRANT SELECT ON frequenta_edizione_insegnamento TO c##studente;
GRANT SELECT ON appello_laurea TO c##studente;
GRANT SELECT ON partecipa_seduta TO c##studente;
GRANT SELECT ON seduta_laurea TO c##studente;
GRANT SELECT ON azienda TO c##studente;
GRANT SELECT ON tutor_aziendale TO c##studente;
GRANT SELECT ON tirocinio TO c##studente;
GRANT SELECT ON appello TO c##studente;
GRANT SELECT ON orario_lezioni TO c##studente;
GRANT SELECT ON seminario TO c##studente;
GRANT SELECT ON presiede_appello TO c##studente;
GRANT SELECT ON ricevimento TO c##studente;
GRANT SELECT ON prenotazione_ricevimento TO c##studente;
GRANT SELECT ON prenotazione_appello TO c##studente;
GRANT SELECT ON partecipa_seminario TO c##studente;
GRANT SELECT ON bando_borsa TO c##studente;
GRANT SELECT ON bando_erasmus TO c##studente;
GRANT SELECT ON assegnazione_borse TO c##studente;
GRANT SELECT ON assegnazione_erasmus TO c##studente;
GRANT SELECT ON email_studente TO c##studente;
GRANT SELECT ON email_docente TO c##studente;
GRANT SELECT ON email_tutor_aziendale TO c##studente;

GRANT SELECT ON vista_bando_borsa_studio TO c##studente;
GRANT SELECT ON vista_bando_erasmus TO c##studente;
GRANT SELECT ON vista_corsi_di_laurea_posti_disponibili TO c##studente;
GRANT SELECT ON vista_seminari_prossimo_mese TO c##studente;

GRANT EXECUTE ON procedura_calcola_statistiche_esami_e_laurea TO c##studente;
```

Come si può notare lo studente ha dei privilegi molto limitati. Ha permessi di *SELECT* su tutte le tabelle. L'unica procedura che può eseguire è quella relativa al calcolo delle statistiche dei suoi esami della laurea. Si fa notare a titolo d'esempio che lo studente non ha i permessi per accedere ai numeri di telefono dei docenti, dei tutor aziendali e di altri studenti. Tutte le operazioni che uno studente deve compiere vanno implementate come procedure apposite così da garantire un maggior grado di sicurezza della base di dati.

Si allegano i privilegi assegnati al docente:

```
GRANT CONNECT, CREATE SESSION TO c##docente;
GRANT SELECT ON corso_laurea TO c##docente;
GRANT SELECT ON studente TO c##docente;
GRANT SELECT ON insegnamento TO c##docente;
GRANT SELECT ON offerta_insegnamento TO c##docente;
GRANT SELECT ON edizione_insegnamento TO c##docente;
GRANT SELECT, INSERT, UPDATE, DELETE ON frequenta_edizione_insegnamento TO c##docente;
GRANT SELECT ON appello_laurea TO c##docente;
GRANT SELECT ON prenotazione_appello_seduta TO c##docente;
GRANT SELECT ON partecipa_seduta TO c##docente;
GRANT SELECT ON seduta_laurea TO c##docente;
GRANT SELECT ON esame_superato TO c##docente;
GRANT SELECT ON relatore TO c##docente;
GRANT SELECT ON questionario TO c##docente;
GRANT SELECT, INSERT, UPDATE ON appello TO c##docente;
GRANT SELECT ON orario_lezioni TO c##docente;
GRANT SELECT, INSERT, UPDATE ON seminario TO c##docente;
GRANT SELECT ON presiede_appello TO c##docente;
GRANT SELECT, INSERT, UPDATE ON ricevimento TO c##docente;
GRANT SELECT ON prenotazione_ricevimento TO c##docente;
GRANT SELECT ON prenotazione_appello TO c##docente;
GRANT SELECT ON telefono_docente TO c##docente;
GRANT SELECT ON email_docente TO c##docente;
GRANT SELECT ON email_studente TO c##docente;
GRANT SELECT ON email_tutor_aziendale TO c##docente;
GRANT SELECT ON docente TO c##docente;
GRANT SELECT ON partecipa_seminario TO c##docente;
GRANT SELECT ON tirocinio TO c##docente;
GRANT SELECT ON azienda TO c##docente;
GRANT SELECT ON tutor_aziendale TO c##docente;
GRANT SELECT ON insegna_edizione TO c##docente;

GRANT SELECT ON vista_corsi_di_laurea_posti_disponibili TO c##docente;
GRANT SELECT ON vista_seminari_prossimo_mese TO c##docente;

GRANT EXECUTE ON procedura_programmazione_appelli TO c##docente;
```

Si fa notare a titolo d'esempio che siccome il docente non ha interesse riguardante i bandi di borsa di studio, di Erasmus, i partecipanti e gli assegnatari, non gli è stato assegnato alcun permesso su tali tabelle. Analogamente il docente non può accedere ai numeri di telefono di studenti e tutor aziendali, né può verificare le tasse addebitate agli studenti.
Tutto sommato il docente ha dei privilegi poco più ampi rispetto allo studente. Per esempio, ha permessi di *SELECT*, inserimento e aggiornamento su *frequenta edizione insegnamento*, su *appello*, su *seminario* e su *ricevimento*. Su tutte le altre tabelle ha solo privilegi di *SELECT*. Infine, ha permessi per eseguire le procedure di programmazione automatizzata degli appelli e dei ricevimenti.

Per quanto riguarda la segreteria, si tratta di un utente comune ma comunque para-amministrativo. Quindi gode di più privilegi e di più ampio respiro. Come si può notare, oltre ad avere i privilegi di *SELECT*, inserimento, aggiornamento e cancellazione su tutte le tabelle, ha privilegi di *SELECT* su tutte le viste, i privilegi per la creazione delle viste e delle procedure, e quelli di esecuzione di tutte le procedure.

Si allegano i privilegi assegnati alla segreteria:

```
GRANT CONNECT, CREATE SESSION TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON corso_laurea TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON studente TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON docente TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON insegnamento TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON edizione_insegnamento TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON offerta_insegnamento TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON insegna_edizione TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON frequenta_edizione_insegnamento TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON appello_laurea TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON prenotazione_appello_seduta TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON partecipa_seduta TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON seduta_laurea TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON relatore TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON azienda TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON tutor_aziendale TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON tirocinio TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON questionario TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON appello TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON orario_lezioni TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON seminario TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON presiede_appello TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON ricevimento TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON esame_superato TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON prenotazione_ricevimento TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON prenotazione_appello TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON tassa TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON partecipa_seminario TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON bando_borsa TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON bando_erasmus TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON partecipazione_bando_borsa TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON partecipazione_bando_erasmus TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON assegnazione_borse TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON assegnazione_erasmus TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON telefono_studente TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON email_studente TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON telefono_tutor_aziendale TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON email_tutor_aziendale TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON telefono_docente TO c##segreteria;
GRANT SELECT, INSERT, UPDATE, DELETE ON email_docente TO c##segreteria;

GRANT SELECT ON vista_corsi_di_laurea_posti_disponibili TO c##segreteria;
GRANT SELECT ON vista_seminari_prossimo_mese TO c##segreteria;
GRANT SELECT ON vista_bando_borsa_studio TO c##segreteria;
GRANT SELECT ON vista_bando_erasmus TO c##segreteria;

GRANT EXECUTE ON procedura_programmazione_appelli TO c##segreteria;
GRANT EXECUTE ON procedura_calcola_statistiche_esami_e_laurea TO c##segreteria;
GRANT EXECUTE ON procedura_assegnazione_tutor_tirocinio TO c##segreteria;
GRANT EXECUTE ON procedura_assegnazione_tutor_docente_tirocinio_interno TO c##segreteria;
GRANT EXECUTE ON procedura_inserimento_insegnamento TO c##segreteria;
GRANT EXECUTE ON procedura_assegnazione_docente_insegna_edizione TO c##segreteria;
```

Scheduler

Lo scheduler è utile per programmare azioni che hanno luogo in istanti predeterminati. Solitamente le azioni che si programmano sono attività periodiche. In maniera del tutto analoga ai trigger, va seguito il costrutto ECA (Event-Condition-Action). L'evento però, è legato allo scorrere del tempo (tempo di sistema) e non ad un'operazione DML come avviene nel caso dei trigger. Esempi di azioni intraprese da uno scheduler in un DB sono manutenzione, pulizia, backup e operazioni di programmazione automatizzate.

Nel caso del DB *esse4,* un'azione utile da automatizzare è legata alla pulizia dei dati e consiste nell'andare ad eliminare le tuple delle prenotazioni degli appelli, le tuple delle prenotazioni degli appelli di laurea e le tuple delle prenotazioni dei ricevimenti nonché dei ricevimenti stessi del mese ormai passato. Tale implementazione viene definita *job.* Qui di seguito, è mostrato il codice per implementare e cancellare il *job.* Si noti come non si effettua alcuna cancellazione di tuple sulla tabella prenotazione ricevimento in maniera esplicita. Infatti, la cancellazione è implicita grazie alle specifiche politiche di reazione adottate.

```
/*
    Job che ogni primo del mese cancella tutti i ricevimenti eseguiti nel mese scorso,
    tutte le prenotazioni ad essi associati, tutte le prenotazioni ad appelli di edizioni
    insegnamento avvenute nel mese scorso e tutte le prenotazioni ad appelli di laurea
    avvenute nel mese scorso.
*/

BEGIN DBMS_SCHEDULER.CREATE_JOB (
    job_name => 'Rollout',
    job_type => 'PLSQL_BLOCK',
    job_action =>  '
                        BEGIN
                            DELETE FROM ricevimento
                            WHERE data_ricevimento < SYSDATE - 30;

                            DELETE FROM prenotazione_appello
                            WHERE data_appello < SYSDATE - 30;

                            DELETE FROM prenotazione_appello_seduta
                            WHERE data_appello < SYSDATE - 30;
                        END;
                    ',
    start_date => TO_DATE('01-SET-2017','DD-MM-YYYY'),
    repeat_interval => 'FREQ = MONTHLY',
    enabled => TRUE,
    comments => 'Cancellazione di tuple di ricevimenti e prenotazioni relative al mese
                 appena passato.');

END;

-- Cancellazione Job di ROLLOUT
BEGIN
    DBMS_SCHEDULER.DROP_JOB ('Rollout');
END;
```