

Relazione Progetto Basi di Dati

Corso di Informatica per il Management

Anna Campagna 0001020141

Altea De Giuseppe 0001028966

Indice

1. Raccolta/Analisi dei requisiti

- *testo completo delle specifiche sui dati,*
- *lista delle operazioni,*
- *tavola media dei volumi,*
- *glossario dei dati*

2. Progettazione Concettuale

- *diagramma E-R,*
- *dizionario delle entità/relazioni,*
- *tavola delle business rules.*

3. Progettazione Logica

- *ristrutturazione dello schema concettuale,*
- *analisi delle ridondanze,*
- *lista delle tabelle con vincoli di chiavi,*
- *lista dei vincoli inter-relazionali.*

4. Descrizione -ad alto livello- delle funzionalità dell'applicazione Web.

5. (In appendice): Codice SQL completo dello schema della base di dati

INTRO

Si vuole realizzare la piattaforma **ESQL** per supportare la didattica del corso di basi di dati, e, nello specifico, del modulo di programmazione SQL.

La piattaforma è liberamente ispirata al tool Moodle SQL utilizzato nelle esercitazioni in aula e contenuto nel sito di Virtuale.

La piattaforma è divisa in due aree, una per i docenti e una per gli studenti.

La prima consente ai docenti diverse funzionalità:

- creare delle tabelle da usare come esercitazione nei test, e di popolarne il contenuto,
- creare test con domande a risposta chiusa o con sketch di codice relative alle tabelle create in precedenza,
- visualizzare le classifiche degli studenti e dei quesiti
- inviare messaggi agli studenti relativamente ad un test.

La parte progettata per gli studenti permette di:

- visionare i test disponibili,
- completare un test e visionarne l'esito,
- inviare messaggi al docente relativamente ad un test.

1) Raccolta e Analisi dei requisiti

1.1) *Testo completo delle specifiche sui dati.*

La piattaforma **ESQL** si basa sul database relazionale **ESQLDB**.

Tutti gli utenti della piattaforma dispongono di: e-mail, nome, cognome, eventuale recapito telefonico.

Gli utenti sono divisi in due tipologie: docenti e studenti.

I primi dispongono anche di: nome del dipartimento di appartenenza e nome del corso di cui sono titolari.

I secondi dispongono anche di un campo anno di immatricolazione e di un codice alfanumerico di lunghezza pari a 16 caratteri.

I docenti possono creare delle tabelle SQL (definite *tabelle di esercizio*, nel seguito): ogni tabella di esercizio dispone di nome, data di creazione, un campo num_righe.

Inoltre, ogni tabella di esercizio dispone di un insieme di attributi: ogni attributo dispone di un nome, un tipo, e può far parte della chiave primaria della tabella di esercizio.

Devono poter essere inseriti dai docenti anche i vincoli di integrità tra attributi di diverse tabelle di esercizio.

In aggiunta, ogni docente può creare dei test: ogni test dispone di un titolo univoco, una data di creazione ed un'eventuale foto.

Ogni test può includere una serie di quesiti: ogni quesito dispone di un numero progressivo (univoco, ma solo all'interno di uno specifico test), un livello di difficoltà (campo enum con valori: Basso, Medio, Alto), un campo descrizione, un campo #numrisposte (ridondanza concettuale, vedere specifiche sotto) e fa riferimento ad una o più tabelle di esercizio tra quelle create dal docente.

I quesiti possono appartenere esclusivamente a due categorie: quesiti a risposta chiusa o quesiti di codice.

Nel primo caso, il quesito dispone di una serie di opzioni di risposta: ogni opzione dispone di una numerazione (univoca, ma solo all'interno di uno specifico quesito) ed un campo testo.

Nel secondo caso, il quesito dispone di una o più soluzioni (sketch di codice SQL che implementano correttamente quanto richiesto dalla descrizione del quesito).

Ogni test dispone di un campo booleano VisualizzaRisposte: se settato a True, le risposte dei quesiti diventano visibili agli studenti, altrimenti restano nascoste.

Gli studenti possono svolgere un test, inserendo una o più risposte per ciascun quesito.

Si vuole tenere traccia del completamento del test, ossia: data di inserimento della prima risposta (su scala temporale), data di inserimento dell'ultima risposta (su scala temporale), stato (campo enum con tre valori: Aperto, InCompletamento, Concluso).

Nel caso di quesiti a risposta chiusa, la risposta consiste nell'opzione scelta tra quelle disponibili.

Nel caso di quesiti di codice, la risposta consiste in un campo testo (codice SQL che risolve l'esercizio).

È prevista la possibilità per lo studente di sottomettere più risposte per lo stesso quesito, in istanti diversi di tempo, ma solo se il test non è stato Concluso.

Ogni risposta dispone di un campo esito, che può valere True o False a seconda che la risposta fornita dallo studente coincida con quella inserita dal docente (nel caso di quesiti a risposta chiusa) o che la risposta fornita dallo studente produca lo stesso output di quella inserita dal docente (nel caso di quesiti di codice).

Infine, è prevista la possibilità di inviare messaggi nella piattaforma.

Ogni messaggio dispone di un titolo, un campo testo, una data di inserimento, e fa riferimento ad uno specifico test.

Il messaggio può essere inviato da un docente: in tal caso, il messaggio viene ricevuto da tutti gli studenti.

Viceversa, un messaggio può essere inviato da uno studente: in tal caso, il destinatario è uno specifico docente.

Infine, si vuole tenere traccia di tutti gli eventi che occorrono nella piattaforma, relativamente all'inserimento di nuovi dati (es. nuovi utenti, nuovi test, nuovi quesiti, etc).

Tali eventi vanno inseriti, sotto forma di messaggi di testo, all'interno di un log, implementato in un' apposita collezione MongoDB.

1.2) Operazioni sui dati.

Operazioni che riguardano tutti gli utenti:

- Autenticazione/registrazione sulla piattaforma
- Visualizzazione dei test disponibili
- Visualizzazione dei quesiti presenti all'interno di ciascun test

Operazioni che riguardano SOLO i docenti:

- Inserimento di una nuova tabella di esercizio, con relativi metadati
- Inserimento di una riga per una tabella di esercizio definita dal docente.

- Creazione di nuovo test
- Creazione di un nuovo quesito con le relative risposte
- Abilitare / disabilitare la visualizzazione delle risposte per uno specifico test
- Inserimento di un messaggio

Operazioni che riguardano SOLO gli studenti:

- Inserimento di una nuova risposta ad un quesito
- Visualizzazione dell'esito della risposta
- Inserimento di un messaggio

Statistiche (visibili da tutti gli utenti):

- Visualizzare la classifica degli studenti, sulla base del numero di test completati (un test si considera completato se il suo stato è pari a "Concluso"). Nella classifica NON devono apparire i dati sensibili dello studente (nome, cognome, e-mail) ma solo il codice alfanumerico.
- Visualizzare la classifica degli studenti, sulla base del numero di risposte corrette inserite rispetto al numero totale di risposte inserite. Nella classifica NON devono apparire i dati sensibili dello studente (nome, cognome, e-mail) ma solo il codice alfanumerico.
- Visualizzare la classifica dei quesiti, in base al numero di risposte inserite dagli studenti.

1.3) Tavola dei Volumi.

- 10 risposte per quesito,
- 20 quesiti,
- 50 utenti.

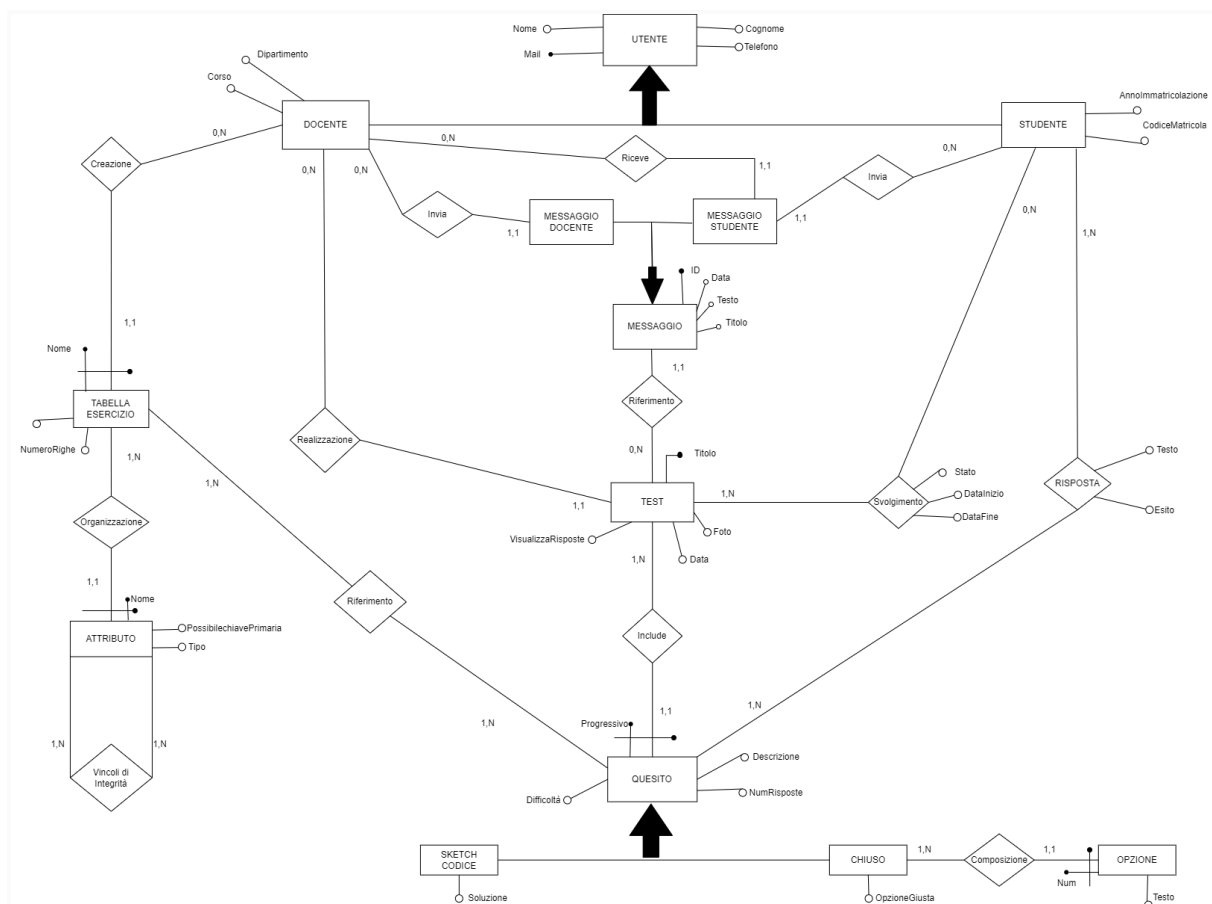
1.4) Glossario dati.

- Gli **utenti** della piattaforma dispongono di: email, nome, cognome, eventuale recapito telefonico e password, sono divisi in docenti e studenti.
- **Docenti**: dispongono anche di nome del dipartimento di appartenenza e nome del corso di cui sono titolari.
- **Studenti**: dispongono anche di un campo anno di immatricolazione e di un codice alfanumerico di lunghezza pari a 16 caratteri.
- **Tabelle di esercizio**: realizzate dal docente per far esercitare gli studenti nei test.
- **Attributo**: compone la tabella di esercizio.
- **Vincoli di integrità referenziale**: devono poter essere inseriti dai docenti tra attributi di diverse tabelle di esercizio.
- **Test**: creato dal docente per farlo eseguire agli studenti, è composto da diversi quesiti.
- **Quesito**: i quesiti possono appartenere esclusivamente a due categorie
 - **Chiuso**: dispone di una serie di opzioni di risposta.

- **Sketch di codice SQL:** lo studente dovrà eseguire un codice sql il cui output verrà confrontato con quello del codice inserito dal docente come soluzione.
- **Svolgimento del test:** appena lo studente apre il test senza compilarlo viene contrassegnato come "Aperto", mentre dal momento che inserirà la prima risposta, passerà a "In completamento"; sarà "Concluso" se verranno date tutte le risposte corrette o il docente decide di chiudere il test rendendo le risposte visibili.
- **Messaggio:** gli utenti possono scambiarsi messaggi facendo riferimento ad uno specifico test.
 - Il messaggio inviato dal docente arriva a tutti gli studenti che svolgono quel test.
 - Il messaggio inviato dallo studente arriverà solo al docente che ha creato il test relativo.

2) Progettazione Concettuale

2.1) Diagramma E-R.



2.2) Dizionario delle entità.

Entità	Descrizione	Attributi	Identificatore
Utente	Persona registrata sulla piattaforma ESQL	Mail, Nome, Cognome, Telefono, Password	Mail
Docente	Insegnante registrato	Corso, Dipartimento	Mail
Studente	Utente che studia	AnnoImmatricolazione, CodiceMatricola	Mail
Test	Raccolta di quesiti, creato dal docente e svolto dallo studente	Titolo, DataTest, Foto, VisualizzaRisposte	Titolo
Quesito	Singola domanda o esercizio all'interno del test	Progressivo, Difficoltà, Descrizione, NumRisposte	Progressivo
Sketch_Codice	Quesito di tipo codice	Soluzione	Progressivo
Quesito_Chiuso	Quesito a risposta chiusa	OpzioneGiusta	Progressivo
Opzione	Scelta di risposta possibile all'interno di un quesito chiuso	Numerazione, Testo	Progressivo, Numerazione
Tabella_Esercizio	Struttura fatta dal docente	Nome, Creazione, NumeroRighe	Nome
Attributo	Elemento di una tabella di esercizio	Nome, Tipo, PossibileChiavePrimaria	Nome, NomeTabella
Messaggio	Comunicazione all'interno della piattaforma da docente a studente e viceversa	Id, Titolo, DataInserimento, Testo	Id

2.3) Dizionario delle relazioni.

Relazione	Descrizione	Componenti	Attributi
Creazione	Il docente crea le tabelle di esercizio	Docente, Tabella_Esercizio	
Organizzazione	Una tabella di esercizio è composta (organizzata) da una serie di attributi	Tabella_Esercizio, Attributo	
Realizzazione	Il docente realizza i test che verranno svolti dagli studenti	Docente, Test	

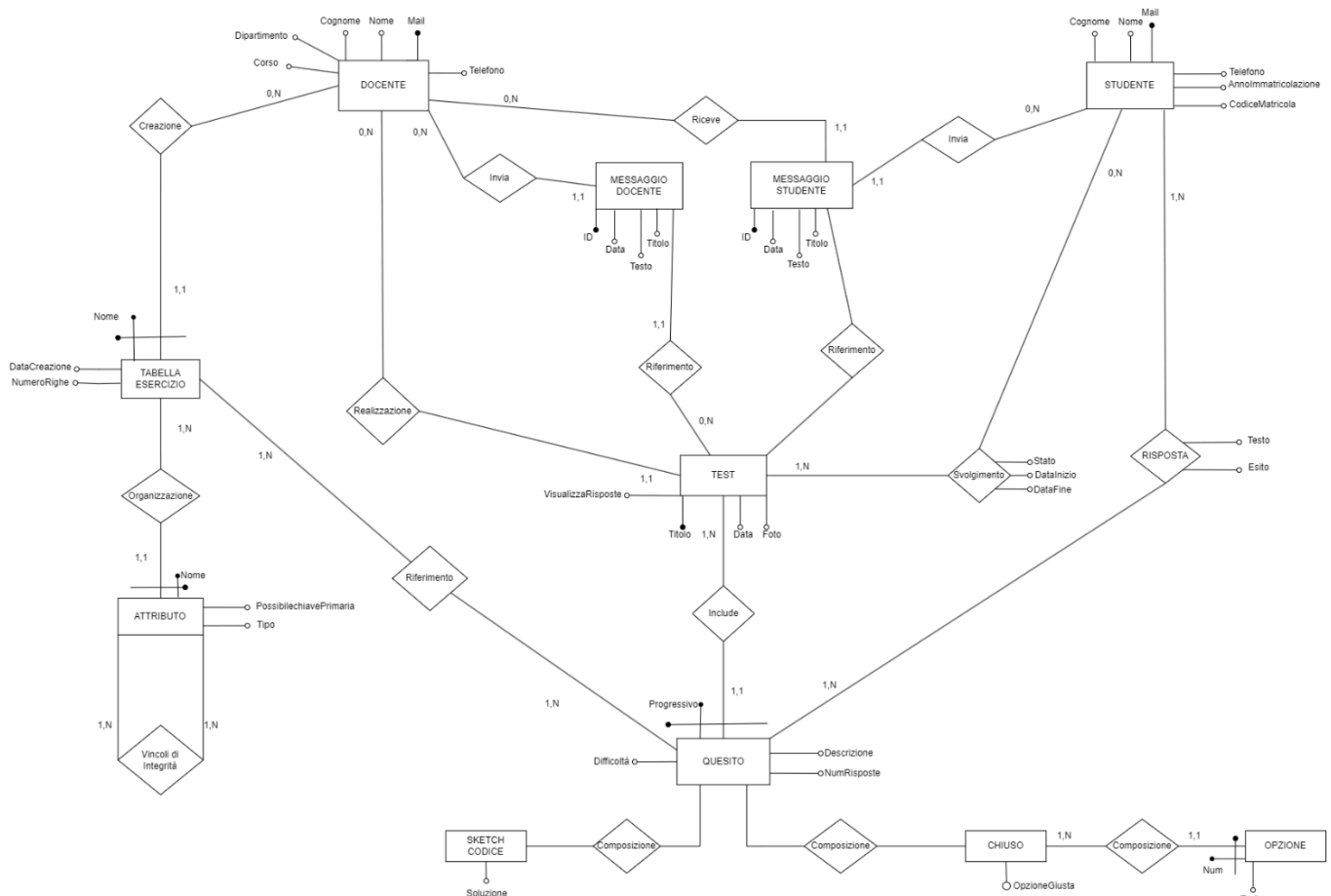
Riferimento	Un quesito fa riferimento ad una o più tabelle di esercizio	Tabella_Esercizio, Quesito	
Svolgimento	Lo studente svolge i test	Studente, Test	DataInizio, DataFine, Stato
Risposta	Lo studente inserisce le risposte ai quesiti	Studente, Quesito	Testo, esito
Include	Il test include una serie di quesiti	Test, Quesito	
Composizione	Il quesito chiuso è composto da almeno due opzioni di risposta	Quesito_Chiuso, Opzione	
Vincolo di integrità	Il docente ha la possibilità di inserire vincoli di integrità tra le chiavi di diverse tabelle	Relazione ricorsiva in attributo	

2.4) Tavola delle business rules.

Business rules
Un UTENTE non può essere sia Docente sia Studente.
STUDENTE.CodiceMatricola è un campo alfanumerico di lunghezza pari a 16 caratteri.
Un QUESITO non può essere sia di Codice sia Chiuso.
QUESITO.Progressivo è univoco, ma solo all'interno dello specifico test.
QUESITO.Difficoltà è un campo enum con valori: Basso, Medio, Alto.
OPZIONE.Num è univoco, ma solo all'interno dello specifico Quesito.
SVOLGIMENTO.Stato è un campo enum con tre valori: Aperto, InCompletamento, Concluso.
L'email deve avere la @ e un punto.

3) Progettazione logica

3.1) Ristrutturazione dello schema concettuale: eliminazione delle generalizzazioni.



Scelta della ristrutturazione.

Nel nostro diagramma abbiamo tre generalizzazioni (utente, messaggio e quesito), tutte e tre totali.

Noi abbiamo scelto l'accorpamento delle entità genitore nelle entità figlie per le entità "utente" e "messaggio", ovvero abbiamo eliminato l'entità madre e i suoi attributi li abbiamo passati alle entità figlie.

Mentre per l'entità "quesito" abbiamo optato per la terza soluzione, quindi abbiamo sostituito la generalizzazione con relazioni tra entità genitore ed entità figlie, "quesito chiuso" e "sketch di codice".

3.2) *Analisi delle ridondanze.*

Operazioni:

- o Aggiungere una nuova risposta ad un quesito esistente (10 volte/mese, interattiva):
- o Rimuovere un quesito e tutte le risposte ottenute (2 volte/mese, batch):
- o Visualizzare tutti gli utenti presenti nella piattaforma (1 volta/mese, batch):
- o Contare il numero di risposte per ciascun quesito presente nella piattaforma (2 volte/mese, interattiva):

Coefficienti

$wI = 1$, $wB = 0.5$, $a = 2$

Tabella dei volumi: 10 risposte per quesito, 20 quesiti, 50 utenti

Il costo delle operazioni con ridondanza sarebbe:

- a. 1 inserimento nella tabella Risposta + 1 update nella tabella Quesito + 1 update nella tabella QuesitoChiuso o SketchCodice

$$C = 10 * 1 * (2 * (1 + 1 + 1) + 0) = 60$$

- b. 1 delete da Quesito + 1 delete da QuesitoChiuso o SketchCodice + 10 delete da Risposta

$$C = 2 * 0.5 * (2 * (1 + 1 + 10) + 0) = 24$$

- c. 50 letture complessive nelle tabelle Studente e Docente

$$C = 1 * 0.5 * (0 + 50) = 25$$

- d. 20 letture in Quesito (1 lettura per 20 quesiti)

$$C = 2 * 1 * (0 + 20) = 40$$

Il costo delle operazioni senza ridondanza sarebbe:

- a. 1 inserimento nella tabella Risposta

$$C = 10 * 1 * (2 * 1 + 0) = 20$$

- b. 1 delete da Quesito + 1 delete da QuesitoChiuso o SketchCodice + 10 delete da Risposta

$$C = 2 * 0.5 * (2 * (1 + 1 + 10) + 0) = 24$$

- c. 50 letture complessive nelle tabelle Studente e Docente

$$C = 1 * 0.5 * (0 + 50) = 25$$

- d. 200 letture in Risposta (10 letture per 20 quesiti)

$$C = 2 * 1 * (0 + 200) = 400$$

Costi totali:

- Totale con ridondanza= 60+24+25+40=149
- Totale senza ridondanza=20+24+25+400=469

469>149

conviene tenere l'attributo NumRisposte

3.3) Schema Logico.

Lista delle tabelle con vincoli di chiavi

- ★ DOCENTE(Mail, Nome, Cognome, Telefono, Corso, Dipartimento)
- ★ STUDENTE(Mail, Nome, Cognome, Telefono, AnnoImmatricolazione, CodiceMatricola)
- ★ TEST(Titolo, DataTest, Foto, VisualizzaRisposte, MailDocente)
- ★ SVOLGIMENTO(MailStudente, TitoloTest, DataInizio, DataFine, Stato)
- ★ QUESITO(Progressivo, TitoloTest, Difficoltà, Descrizione, NumRisposte)
- ★ SKETCH_CODICE(Progressivo, TitoloTest, Soluzione)
- ★ QUESITO_CHIUSO(Progressivo, TitoloTest, OpzioneGiusta)
- ★ OPZIONE(Numerazione, ProgressivoChiuso, TitoloTest, Testo)
- ★ RISPOSTA(ProgressivoQuesito, TitoloTest, MailStudente, Esito, Testo)
- ★ TABELLA_ESERCIZIO(Nome, Creazione, NumeroRighe, MailDocente)
- ★ RIF_TABELLA_QUESITO(Progressivo, TitoloTest, NomeTabella)
- ★ ATTRIBUTO(Nome, NomeTabella, Tipo, PossibileChiavePrimaria)
- ★ VINCOLO(NomeAttributoPK, NomeTabellaPK, NomeAttributoFK, NomeTabellaFK)
- ★ MESSAGGIODOCENTE(Id, TitoloMess, Testo, DataInserimento, TitoloTest, MailDocente)
- ★ MESSAGGIOSTUDENTE(Id, TitoloMess, Testo, DataInserimento, TitoloTest, MailStudente, MailDocente)

Lista dei vincoli di integrità referenziale

TEST.MailDocente → DOCENTE.Mail

SVOLGIMENTO.MailStudente → STUDENTE.Mail

SVOLGIMENTO.TitoloTest → TEST.Titolo

QUESITO.TitoloTest → TEST.Titolo

SKETCH_CODICE.Progressivo → QUESITO.Progressivo

SKETCH_CODICE.TitoloTest → QUESITO.TitoloTest

QUESITO_CHIUSO.Progressivo → QUESITO.Progressivo

QUESITO_CHIUSO.TitoloTest → QUESITO.TitoloTest

OPZIONE.ProgressivoChiuso → QUESITO_CHIUSO.Progressivo
OPZIONE.TitoloTest → QUESITO_CHIUSO.TitoloTest

RISPOSTA.ProgressivoQuesito → QUESITO.Progressivo
RISPOSTA.TitoloTest → QUESITO.TitoloTest
RISPOSTA.MailStudente → STUDENTE.Mail

TABELLA_ESERCIZIO.MailDocente → DOCENTE.Mail

RIF_TABELLA_QUESITO.Progressivo, TitoloTest → QUESITO.Progressivo, TitoloTest
RIF_TABELLA_QUESITO.NomeTabella → TABELLA_ESERCIZIO.Nome

ATTRIBUTO.NomeTabella → TABELLA_ESERCIZIO.Nome

VINCOLO.NomeAttributoPK → ATTRIBUTO.Nome
VINCOLO.NomeTabellaPK → ATTRIBUTO.NomeTabella
VINCOLO.NomeAttributoFK → ATTRIBUTO.Nome
VINCOLO.NomeTabellaFK → ATTRIBUTO.NomeTabella

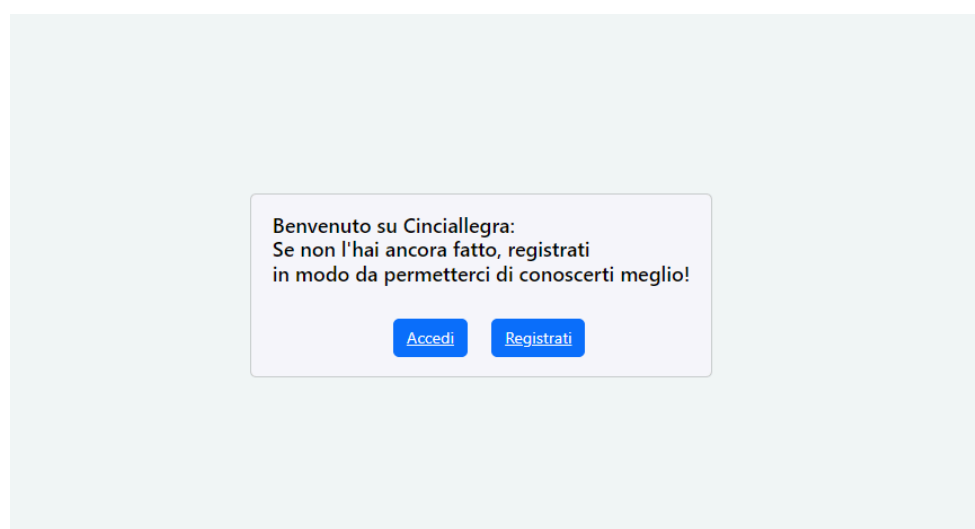
MESSAGGIODOCENTE.TitoloTest → TEST.Titolo
MESSAGGIODOCENTE.MailDocente → DOCENTE.Mail

MESSAGGIOSTUDENTE.TitoloTest → TEST.Titolo
MESSAGGIOSTUDENTE.MailDocente → DOCENTE.Mail
MESSAGGIOSTUDENTE.MailStudente → STUDENTE.Mail

4) Descrizione delle funzionalità dell'applicazione web

HOME PAGE

L'homepage è la prima interfaccia che viene visualizzata dall'utente. Qui egli può decidere di registrarsi sulla piattaforma, oppure accedere se è già registrato.



REGISTRAZIONE

Un nuovo utente che apre per la prima volta la piattaforma dovrà registrarsi, selezionando dapprima il suo status (docente o studente) ed inserendo poi i dati necessari al fine della registrazione, quali:

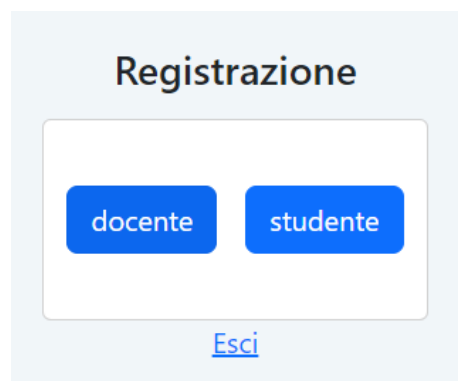
- Nome
- Cognome
- Email
- Telefono

Se è uno studente inserirà anche

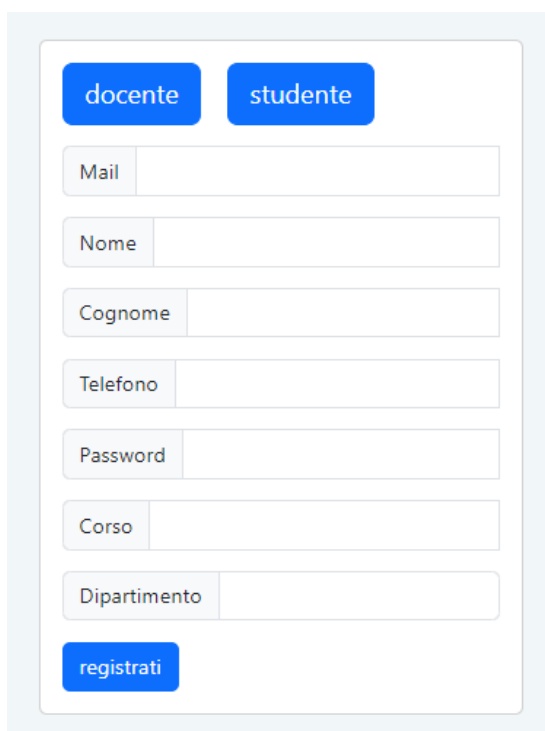
- Anno di iscrizione
- Codice matricola

Se è un docente, invece, inserirà anche

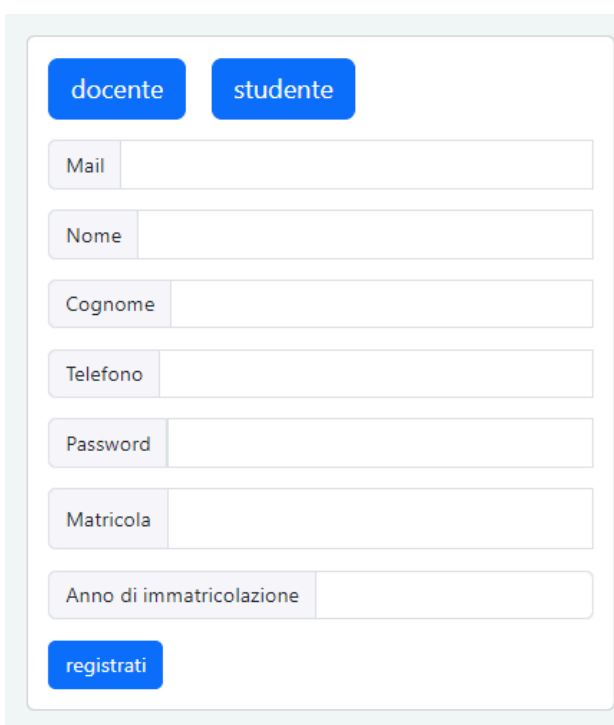
- Dipartimento
- Corso



Registration screen titled "Registrazione". It features two blue buttons: "docente" and "studente". Below these buttons is a blue link labeled "Esci".



Registration form for teachers. It includes a selection of "docente" or "studente". The form fields are: Mail, Nome, Cognome, Telefono, Password, Corso, and Dipartimento. A blue "registri" button is at the bottom.



Registration form for students. It includes a selection of "docente" or "studente". The form fields are: Mail, Nome, Cognome, Telefono, Password, Matricola, and Anno di immatricolazione. A blue "registri" button is at the bottom.

Una volta verificato che tutti i campi siano stati completati e che non esista un altro utente con la stessa email, l'utente verrà registrato nel database e potrà accedere con le sue credenziali.

Registrazione avvenuta con successo!

Accedi

Altrimenti, se esiste un altro utente con lo stesso indirizzo mail (quindi l'utente è già registrato), si chiederà di eseguire l'accesso.

Esiste già un utente con questa email,
[Accedi!!!!](#)

ACCESSO

Se l'utente si è già registrato precedentemente, può inserire le credenziali (email e password) per accedere alla piattaforma. Le credenziali verranno controllate e, una volta accertato che il profilo esista, si aprirà la pagina del sito in base al tipo di utente che ha eseguito l'accesso (studente o docente).

Accedi

[Esci](#)

Nel caso in cui l'utente non esistesse nel database o la password fosse sbagliata, viene inviato un messaggio di errore chiedendo eventualmente di registrarsi.

Utente non trovato,
[Registrati!!!!](#)
(O forse hai solo sbagliato password)


LOGOUT

Una volta eseguito l'accesso, indipendentemente se si è uno studente o un docente, si può eseguire il logout che riporta alla homepage iniziale.

HOMEPAGE DOCENTE

Il docente dalla sua home page ha la possibilità di scegliere cosa fare tra le seguenti attività:


- creare tabelle da utilizzare nei test come esercitazione
- popolare le tabelle create
- creare vincoli di integrità fra le tabelle create
- creare dei test e i quesiti al suo interno
- visualizzare i test da lui creati
- mandare agli studenti messaggi relativamente ad uno specifico test e visualizzare i messaggi che i suoi studenti hanno inviato relativamente allo stesso
- visualizzare le classifiche riguardanti gli studenti e i quesiti

 Cinciallegra

benvenuto/a alte

[Tabelle](#) [Crea test](#) [Visualizza classifiche](#) [Visualizza test](#) [Logout](#)

(se si clicca il bottone “Tabelle”)

 Cinciallegra

benvenuto/a alte

[Tabelle](#) [Crea test](#) [Visualizza classifiche](#) [Visualizza test](#) [Logout](#)

[Crea un nuova tabella](#) [Popola una tabella](#) [Crea vincoli di integrità](#)

CREAZIONE DELLA TABELLA

Nella pagina dedicata alla creazione delle tabelle, viene mostrato un form nel quale viene richiesto inizialmente di inserire il nome della tabella che si desidera creare e dopodichè si potrà scegliere il numero di attributi che la compongono.

 **CREA TABELLE**

Inserisci il nome della tabella

[Inserisci nome](#)

CREA TABELLE

Inserisci il nome della tabella

Inserisci nome

Inserisci il numero di colonne della tabella

Seleziona numero colonne

Scelto il numero di attributi si aprirà una facciata con dei campi da riempire con il nome dell'attributo, il tipo dello stesso e se è una chiave primaria oppure no, per ciascuno degli n attributi.

inserisci nome attributo

inserisci nome attributo

inserisci nome attributo

inserisci tipo

inserisci tipo

VARCHAR(10)
VARCHAR(50)
VARCHAR(100)
INT
BOOLEAN
DOUBLE

Conferma attributi

è chiave primaria?

è chiave primaria?

è chiave primaria?

CREAZIONE DEI VINCOLI DI INTEGRITA' REFERENZIALE

Una volta create almeno due tabelle il docente sceglierà quali tabelle avranno un vincolo di integrità referenziale.

VINCOLI DI INTEGRITA' REFERENZIALE

Scegli tabella

Scelta la prima tabella su cui si vuole imporre il vincolo compariranno solo gli attributi chiave di questa, a cui farà riferimento la seconda tabella.

Poi comparirà una tendina con il nome di tutte le altre tabelle esistenti (fuorchè la prima selezionata).

hai selezionato la tabella citta

nome VARCHAR(50)
stato VARCHAR(50)

 ▼

Scegli tabella

Scelta anche la seconda tabella compare una schermata che mette in riga l'attributo chiave della prima con la possibilità di scegliere quale attributo della seconda tabella fa riferimento ad esso.

citta

viaggio

nome

 ▼

stato

 ▼

Scegli attributi

Infine, cliccando "Scegli attributi" viene stampato il messaggio di creazione del vincolo con la possibilità di creare un ulteriore vincolo.

creazione vincolo fisico effettuato

Crea vincolo

POPOLAMENTO DELLA TABELLA

In questa pagina si presenta un menù a tendina con la lista delle tabelle create dal docente che sta utilizzando la piattaforma.

POPOLAMENTO TABELLE

 ▼
 ▼
citta
Scegli

Si seleziona la tabella che si vuole popolare inserendo una nuova riga e si compilano i campi relativi agli attributi presenti in quella tabella con i valori desiderati.

Tabella: citta

nome (VARCHAR(50)):

stato (VARCHAR(50)):

numAbitanti (INT):

Invia

CREAZIONE DEL TEST

Per prima cosa, in questa facciata, si inserirà obbligatoriamente il titolo del test che si vuole creare. Il docente ha anche la possibilità di caricare una foto riguardante quel test. La foto deve trovarsi nella cartella img, all'interno della cartella in cui sono collocati tutti i sorgenti del progetto.

CREAZIONE DI UN NUOVO TEST

Titolo del Test:

Scegli immagine: (facoltativo)

Scegli file Nessun file selezionato

Crea Test

Una volta creato il test, è possibile andare a creare i quesiti che si troveranno al suo interno, cliccando il bottone "Aggiungi Quesito" che comparirà sotto.

Test creato.

Inserisci i quesiti

Aggiungi Quesito

CREAZIONE DEI QUESITI PER IL TEST

I quesiti possono essere creati uno alla volta, ma devono essere realizzati tutti nel momento della creazione del test.

CREA QUESITO

Descrizione quesito

Difficoltà quesito: ☐ Basso ☐ Medio ☐ Alto

Tabella di riferimento:

Tipo di risposta: ☐ Quesito chiuso ☐ Sketch di codice

Crea

Si inizia inserendo il testo del quesito, poi tramite dei radio buttons si sceglie la difficoltà, si sceglie a quale tabella esercizio fa riferimento questo quesito e infine si sceglie che tipo di quesito è:

- Chiuso, bisognerà inserire le tre risposte per le opzioni indicando qual è quella corretta

Tipo di risposta: ☒ Quesito chiuso ☐ Sketch di codice

Opzione 1: ☐ opzione giusta

Opzione 2: ☐ opzione giusta

Opzione 3: ☐ opzione giusta

Crea

- Sketch di codice, dove si inserisce la query o il testo corretto.

Tipo di risposta: ☐ Quesito chiuso ☒ Sketch di codice

inserisci la soluzione:

Crea

VISUALIZZAZIONE DEI TEST

Se il docente vuole, può visualizzare la lista dei test che ha creato (con a fianco la data di creazione).

VISUALIZZA TEST

- [Test1 \(2024-05-30 13:12:45\)](#)
- [TestNuovo \(2024-05-30 13:15:42\)](#)

Se il docente clicca su uno dei titoli si aprirà la pagina relativa a quel test dove può visionare i quesiti con il rispettivo livello di difficoltà.

Può decidere anche se chiudere il test per tutti gli studenti selezionando la checkbox “Visualizza risposte” in modo da mostrare tutte le risposte corrette agli studenti.

O anche eliminare il test.

DETTAGLI TEST [Inserisci messaggio](#) [Messaggi ricevuti](#)

Test1

1) Cosa vuol dire CRUD?

Livello: Medio

2) Seleziona tutti gli elementi di città

Livello: Basso

Visualizza risposte ☐

Elimina test ☐

Conferma

Inoltre, in questa schermata il docente ha la possibilità di inviare dei messaggi relativi allo specifico test selezionato (i quali saranno visualizzati dagli studenti che apriranno quel test) e di leggere tutti i messaggi che gli studenti hanno inviato al docente riguardo quello stesso test.

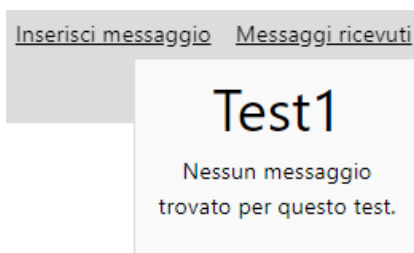
INVIO MESSAGGIO DOCENTE



The form is titled "Inserisci messaggio" and "Messaggi ricevuti". It contains two text input fields: "Oggetto del messaggio:" and "Testo del messaggio:". Below the second field is a dark blue button labeled "Invia messaggio".

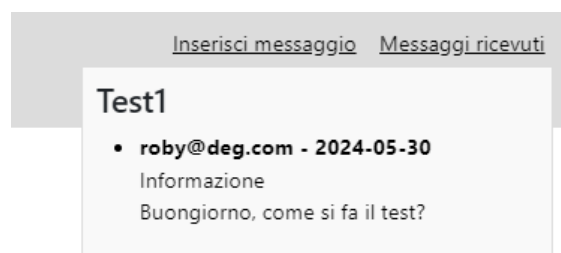
Se si clicca "Inserisci messaggio", compare un piccolo form con due campi testo nei quali viene richiesto di inserire l'oggetto del messaggio che il docente desidera inviare agli studenti, e il testo del messaggio stesso. Occorre premere il bottone "Invia messaggio" affinché il messaggio venga inviato correttamente.

Se si clicca "Messaggi ricevuti" si aprirà un piccolo banner nel quale verranno visualizzati il titolo del test cui si riferiscono i messaggi ricevuti e l'elenco dei messaggi. In particolare verranno visualizzati la mail dello studente che ha inviato il messaggio, la data in cui è stato inviato, l'oggetto e il corpo del messaggio.



The banner is titled "Inserisci messaggio" and "Messaggi ricevuti". It displays the title "Test1" and the message "Nessun messaggio trovato per questo test."

(in questo caso non c'è nessun messaggio)



The banner is titled "Inserisci messaggio" and "Messaggi ricevuti". It displays the title "Test1" and a list of messages. The first message is from "robby@deg.com" dated "2024-05-30" with the subject "Informazione" and the body "Buongiorno, come si fa il test?".

(in questo caso sono presenti dei messaggi)

VISUALIZZAZIONE CLASSIFICHE

In questa pagina (accessibile sia dagli studenti sia dai docenti) l'utente può visualizzare una serie di statistiche, nello specifico 3 classifiche: la prima è la classifica degli studenti sulla base del maggior numero di test completati; la seconda è la classifica degli studenti sulla base della percentuale di risposte corrette inserite nello svolgimento dei test presenti sulla piattaforma; la terza è la classifica dei quesiti sulla base del maggior numero di risposte inserite agli stessi.

CLASSIFICHE		
Classifica per numero di test completati: <i>Nessun risultato trovato per la classifica dei test completati.</i>	Classifica per percentuale di risposte corrette: <i>Nessun risultato trovato per la classifica delle risposte corrette.</i>	Classifica dei quesiti: 1. Descrizione quesito: Cosa vuol dire CRUD? - Numero risposte: 0 2. Descrizione quesito: Seleziona tutti gli elementi di città - Numero risposte: 0 3. Descrizione quesito: domandona da mille dollari - Numero risposte: 0


CLASSIFICHE		
Classifica per numero di test completati: 1. Codice studente: 1234567890123123 - Test completati: 1	Classifica per percentuale di risposte corrette: 1. Codice studente: 1234567890123756 - Risposte giuste: 1.0000 2. Codice studente: 1234567890123123 - Risposte giuste: 0.0000	Classifica dei quesiti: 1. Descrizione quesito: Cosa vuol dire CRUD? - Numero risposte: 2 2. Descrizione quesito: come vaaa - Numero risposte: 1 3. Descrizione quesito: Seleziona tutti gli elementi dalla tabella città - Numero risposte: 1 4. Descrizione quesito: domandaaaa - Numero risposte: 0

HOMEPAGE STUDENTE


Lo studente può accedere alla piattaforma nello stesso modo del docente, poiché la stored procedure di autenticazione si occuperà di distinguere il tipo di utente.

Una volta eseguito l’accesso si apre la home page per lo studente dove questo può:

- visualizzare i test da svolgere
- visualizzare le classifiche

 Cinciallegra	benvenuto/a anna
<div>Visualizza test Classifiche Logout</div>	

Se si clicca “visualizza test”

 Cinciallegra	benvenuto/a anna
<div>Visualizza test Classifiche Logout</div>	
<div>Test1 (2024-05-30 17:34:45, alte@deg.it)</div> <div>Test2 (2024-05-30 17:35:29, alte@deg.it)</div>	

SVOLGIMENTO TEST

Dopo aver cliccato “Visualizza test” c’è la possibilità di scegliere un test da svolgere. Cliccando il titolo si aprirà la relativa pagina per poter rispondere ai quesiti.

Lo studente ha la possibilità di uscire ed entrare dal test e dare le risposte che vuole, tutte le volte che vuole finché non risponderà correttamente a tutti i quesiti oppure finché il docente non deciderà di concludere il test.

Sotto ogni test possiamo individuare il suo stato:
sarà Aperto dal momento che lo studente lo apre senza compilarlo, InCompletamento se inserisce la prima risposta, Concluso se lo studente risponde correttamente a tutte le domande o il docente decide di chiudere il test.

[Visualizza test](#)[Classifiche](#)[Logout](#)[Test1](#)

(2024-05-30 17:34:45, alte@deg.it)

Stato: Aperto

[Test2](#)

(2024-05-30 17:35:29, alte@deg.it)

Stato: InCompletamento

INVIO MESSAGGIO STUDENTE

Sempre all'interno della pagina di svolgimento test lo studente può inviare al docente che ha creato il test un messaggio e leggere quelli inviati dallo stesso professore.

SVOLGI IL TEST**Test1**[Inserisci messaggio](#) [Messaggi ricevuti](#)**Test1**

- alte@deg.it - 2024-05-30
Male
Ragazzi avete sbagliato tutto

MONGODB

Si vuole tenere traccia di tutti gli eventi che occorrono nella piattaforma, relativamente all'inserimento di nuovi dati (es. nuovi utenti, nuovi test, nuovi quesiti, etc). Tali eventi vanno inseriti, sotto forma di messaggi di testo, all'interno di un log, implementato in un'apposita collezione MongoDB.

```
_id: ObjectId('6659bd632f17632455096ee6')  
message: "Nuovo test Test1 inserito"  
timestamp: 2024-05-31T12:06:59.000+00:00
```

```
_id: ObjectId('6659bde42f17632455096ee7')  
message: "Nuovo quesito 1 inserito"  
timestamp: 2024-05-31T12:09:08.000+00:00
```

```
_id: ObjectId('6659bde42f17632455096ee8')  
message: "Nuova opzione Crush, Refresh, Update, Delay inserita"  
timestamp: 2024-05-31T12:09:08.000+00:00
```

```
_id: ObjectId('6659be922f17632455096ee9')  
message: "Nuovo quesito 1 inserito"  
timestamp: 2024-05-31T12:12:02.000+00:00
```

CODICE SQL

Tabelle

```
DROP DATABASE IF EXISTS cincialleggra;
DROP DATABASE IF EXISTS moodle;
CREATE DATABASE cincialleggra;
USE cincialleggra;

/* Tabella del Docente */
create table DOCENTE(
/* Attributi comuni che hanno entrambi gli utenti */
    Mail VARCHAR(40) PRIMARY KEY,
    Nome VARCHAR(40),
    Cognome VARCHAR(40) ,
    Telefono BIGINT,
    Pass VARCHAR(40) ,

/* Attributi speciali di cui solo il Docente dispone*/
    Corso VARCHAR(30),
    Dipartimento VARCHAR(60)
) engine=INNODB;

/* Tabella dello Studente */
create table STUDENTE(
/* Attributi comuni che hanno entrambi gli utenti */
    Mail VARCHAR(40) PRIMARY KEY,
    Nome VARCHAR(40),
    Cognome VARCHAR(40) ,
    Telefono BIGINT,
    Pass VARCHAR(40) ,

/* Attributi speciali di cui solo lo Studente dispone*/
    AnnoImmatricolazione BIGINT,
    /* Attenzione deve essere un codice alfanumerico di lunghezza pari a 16 caratteri */
    CodiceMatricola VARCHAR(16),
    CONSTRAINT FORMATO_Codice CHECK( LENGTH(CodiceMatricola) = 16)
) engine=INNODB;

/* Tabella del Test*/
create table TEST(
    Titolo VARCHAR(30) PRIMARY KEY,
    DataTest datetime,
    Foto VARCHAR(200), /*la foto è eventuale */
    VisualizzaRisposte tinyint, /* se true gli studenti possono vederle, altrimenti se settato a false non possono.*/
    MailDocente VARCHAR(40),

    FOREIGN KEY (MailDocente) REFERENCES DOCENTE(Mail) ON DELETE CASCADE
) engine=INNODB;
```



```

create table SVOLGIMENTO(
    MailStudente VARCHAR(40),
    TitoloTest VARCHAR(30),
    DataInizio datetime,
    DataFine datetime,
    Stato ENUM('Aperto','InCompletamento','Concluso') DEFAULT 'Aperto',

    PRIMARY KEY (MailStudente, TitoloTest),
    FOREIGN KEY (MailStudente) REFERENCES STUDENTE(Mail) ON DELETE CASCADE,
    FOREIGN KEY (TitoloTest) REFERENCES TEST(Titolo) ON DELETE CASCADE
) engine=INNODB;

create table QUESITO(
    Progressivo INT,
    TitoloTest VARCHAR(30),
    Difficolta ENUM ('Basso','Medio','Alto'),
    Descrizione VARCHAR(400),
    NumRisposte INT,

    PRIMARY KEY (Progressivo, TitoloTest),
    FOREIGN KEY (TitoloTest) REFERENCES TEST(Titolo) ON DELETE CASCADE
) engine=INNODB;

create table SKETCH_CODICE( /* Ha senso differenziarli perché derivano da una generalizzazione Totale di quesito. */
    Progressivo INT,
    TitoloTest VARCHAR(30),

    Soluzione VARCHAR(300), /*La soluzione è la query come richiesta dal docente*/

    PRIMARY KEY (Progressivo, TitoloTest),
    FOREIGN KEY (Progressivo) REFERENCES QUESITO(Progressivo) ON DELETE CASCADE,
    FOREIGN KEY (TitoloTest) REFERENCES QUESITO(TitoloTest) ON DELETE CASCADE
) engine=INNODB;

create table QUESITO_CHIUSO(
    Progressivo INT,
    TitoloTest VARCHAR(30),

    OpzioneGiusta VARCHAR(1), /*L'opzione giusta è l'opzione segnata come esatta dal docente: a, b, c...*/

    PRIMARY KEY (Progressivo, TitoloTest),
    FOREIGN KEY (Progressivo) REFERENCES QUESITO(Progressivo) ON DELETE CASCADE,
    FOREIGN KEY (TitoloTest) REFERENCES QUESITO(TitoloTest) ON DELETE CASCADE
) engine=INNODB;

```

```

create table OPZIONE(
    Numerazione INT,
    ProgressivoChiuso INT,
    TitoloTest VARCHAR(30),

    Testo VARCHAR(40), /* Testo della singola opzione. */

    PRIMARY KEY (Numerazione, ProgressivoChiuso, TitoloTest),
    FOREIGN KEY (ProgressivoChiuso) REFERENCES QUESITO_CHIUSO(Progressivo) ON DELETE CASCADE,
    FOREIGN KEY (TitoloTest) REFERENCES QUESITO_CHIUSO(TitoloTest) ON DELETE CASCADE
) engine=INNODB;

create table RISPOSTA(
    ProgressivoQuesito INT,
    TitoloTest VARCHAR(30),
    MailStudente VARCHAR(40),

    Esito boolean, /* Campi di Risposta. */
    Testo VARCHAR(300),

    PRIMARY KEY(ProgressivoQuesito, TitoloTest, MailStudente),
    FOREIGN KEY (ProgressivoQuesito) REFERENCES QUESITO(Progressivo) ON DELETE CASCADE,
    FOREIGN KEY (TitoloTest) REFERENCES QUESITO(TitoloTest) ON DELETE CASCADE,
    FOREIGN KEY (MailStudente) REFERENCES STUDENTE(Mail) ON DELETE CASCADE
) engine=INNODB;

/* La tabella dell'esercizio si riferisce:
   Al DOCENTE che la crea (una Tabella di Esercizio può essere creata soltanto da un docente). */
create table TABELLA_ESERCIZIO(
    Nome VARCHAR(30) ,
    Creazione DATE,
    NumeroRighe INT,
    MailDocente VARCHAR(40),

    PRIMARY KEY(Nome, MailDocente),
    FOREIGN KEY (MailDocente) REFERENCES DOCENTE(Mail) ON DELETE CASCADE
) engine=INNODB;

create table RIF_TABELLA_QUESITO(
    ProgressivoQuesito INT,
    TitoloTest VARCHAR(30),
    NomeTabella VARCHAR(30),
    MailDocente VARCHAR(40),

    PRIMARY KEY(ProgressivoQuesito, TitoloTest, NomeTabella, MailDocente),
    FOREIGN KEY (ProgressivoQuesito, TitoloTest) REFERENCES QUESITO(Progressivo, TitoloTest) ON DELETE CASCADE,
    FOREIGN KEY (NomeTabella, MailDocente) REFERENCES TABELLA_ESERCIZIO(Nome, MailDocente) ON DELETE CASCADE
) engine=INNODB;

```

```

create table ATTRIBUTO( /* Tabella relativa agli Attributi. */
    NomeTabella VARCHAR(30),
    Nome VARCHAR(30),
    Tipo VARCHAR(30),

    /* Ogni ATTRIBUTO può fare parte della Chiave Primaria della TABELLA_ESERCIZIO. */
    PossibileChiavePrimaria boolean NOT NULL,

    PRIMARY KEY (Nome, NomeTabella),
    FOREIGN KEY (NomeTabella) REFERENCES TABELLA_ESERCIZIO(Nome) ON DELETE CASCADE
) engine=INNODB;

create table VINCOLO(
    NomeAttributoPK VARCHAR(30),
    NomeTabellaPK VARCHAR(30),
    NomeAttributoFK VARCHAR(30),
    NomeTabellaFK VARCHAR(30),

    PRIMARY KEY (NomeAttributoPK, NomeTabellaPK, NomeAttributoFK, NomeTabellaFK),
    FOREIGN KEY (NomeAttributoPK) REFERENCES ATTRIBUTO(Nome) ON DELETE CASCADE,
    FOREIGN KEY (NomeTabellaPK) REFERENCES ATTRIBUTO(NomeTabella) ON DELETE CASCADE,
    FOREIGN KEY (NomeAttributoFK) REFERENCES ATTRIBUTO(Nome) ON DELETE CASCADE,
    FOREIGN KEY (NomeTabellaFK) REFERENCES ATTRIBUTO(NomeTabella) ON DELETE CASCADE
) engine=INNODB;

create table MESSAGGIODOCENTE(
    Id INT PRIMARY KEY AUTO_INCREMENT,
    TitoloMess VARCHAR(30), /* Rappresenta l'oggetto della mail ad esempio.*/
    Testo VARCHAR(100),
    DataInserimento date,
    TitoloTest VARCHAR(30),
    MailDocente VARCHAR(40),

    FOREIGN KEY (TitoloTest) REFERENCES TEST(Titolo) ON DELETE CASCADE,
    FOREIGN KEY (MailDocente) REFERENCES DOCENTE(Mail) ON DELETE CASCADE
) engine=INNODB;

create table MESSAGGIOSTUDENTE(
    Id INT PRIMARY KEY AUTO_INCREMENT,
    TitoloMess VARCHAR(30), /* Rappresenta l'oggetto della mail ad esempio.*/
    Testo VARCHAR(100),
    DataInserimento date,
    TitoloTest VARCHAR(30),
    MailStudente VARCHAR(40),
    MailDocente VARCHAR(40),

    FOREIGN KEY (TitoloTest) REFERENCES TEST(Titolo) ON DELETE CASCADE,
    FOREIGN KEY (MailDocente) REFERENCES DOCENTE(Mail) ON DELETE CASCADE,
    FOREIGN KEY (MailStudente) REFERENCES STUDENTE(Mail) ON DELETE CASCADE
) engine=INNODB;

```

Stored procedure

```
/* OPERAZIONI SUI DATI */
use cinciallegra;

/*-----*/
/*OPERAZIONI che riguardano TUTTI GLI UTENTI:*/
/* 1) autenticazione sulla piattaforma. */
/* who è 1 per docente e 2 per studente*/
DELIMITER $
CREATE PROCEDURE Autenticazione(IN InputMail VARCHAR(30), IN Pass VARCHAR(30), OUT who INT)
) BEGIN
    DECLARE countDocenti INT DEFAULT 0;
    DECLARE countStudenti INT DEFAULT 0;

    SELECT COUNT(*) INTO countDocenti FROM DOCENTE WHERE docente.Mail = InputMail AND DOCENTE.Pass=Pass;
    SELECT COUNT(*) INTO countStudenti FROM STUDENTE WHERE studente.Mail = InputMail AND studente.Pass=Pass;

    IF countDocenti = 1 THEN
        SET who = 1;
    ELSEIF countStudenti = 1 THEN
        SET who = 2;
    END IF;
END$
DELIMITER ;

DELIMITER $ /* 1) registrazione sulla piattaforma*/
CREATE PROCEDURE RegistrazioneStudiante(IN Mail VARCHAR(30), IN Nome VARCHAR(30), IN Cognome VARCHAR(30), IN Telefono BIGINT, IN Pass VARCHAR(30), IN Matricola VARCHAR(16),
IN AnnoImmatricolazione BIGINT, out registrazione BOOLEAN)
) BEGIN
    DECLARE countDocenti INT DEFAULT 0;
    DECLARE countStudenti INT DEFAULT 0;

    SET countDocenti = (SELECT COUNT(*) FROM STUDENTE WHERE (Mail = STUDENTE.Mail));
    SET countStudenti = (SELECT COUNT(*) FROM DOCENTE WHERE (Mail = DOCENTE.Mail));

    IF(countDocenti = 0) AND (countStudenti = 0) THEN
        INSERT INTO STUDENTE VALUES (Mail, Nome, Cognome, Telefono, Pass, AnnoImmatricolazione, Matricola);
        set registrazione=true;
    END IF;
END$

DELIMITER $ /* 1) registrazione sulla piattaforma*/
CREATE PROCEDURE RegistrazioneDocente(IN Mail VARCHAR(30), IN Nome VARCHAR(30), IN Cognome VARCHAR(30), IN Telefono BIGINT, IN Pass VARCHAR(30), IN Corso VARCHAR(30),
IN Dipartimento VARCHAR(30), out registrazione BOOLEAN)
) BEGIN
    DECLARE countDocenti INT DEFAULT 0;
    DECLARE countStudenti INT DEFAULT 0;

    SET countDocenti = (SELECT COUNT(*) FROM STUDENTE WHERE (Mail = STUDENTE.Mail));
    SET countStudenti = (SELECT COUNT(*) FROM DOCENTE WHERE (Mail = DOCENTE.Mail));

    IF(countDocenti = 0) AND (countStudenti = 0) THEN
        INSERT INTO DOCENTE VALUES (Mail, Nome, Cognome, Telefono, Pass, Corso, Dipartimento);
        set registrazione=true;
    END IF;
END$

DELIMITER $ /* 3) Visualizzazione dei quesiti presenti all'interno di ciascun test. */
CREATE PROCEDURE VisualizzazioneQuesiti(IN Titolo VARCHAR(30))
) BEGIN
    SELECT *
    FROM QUESITO
    WHERE TitoloTest = Titolo;
END$
```

```

DELIMITER $ /* 2) Visualizzazione dei test disponibili creati da un determinato docente. */
CREATE PROCEDURE VisualizzazioneTestDoc(IN Mail VARCHAR(40))
BEGIN
    SELECT *
    FROM TEST
    WHERE MailDocente = Mail;
END$

DELIMITER $ /* 2) Visualizzazione di tutti i test disponibili. (per lo studente) */
CREATE PROCEDURE VisualizzazioneTest(IN Mail VARCHAR(30))
BEGIN

    SELECT t.Titolo, t.DataTest, t.MailDocente, s.Stato
    FROM TEST as t
    LEFT JOIN svolgimento as s
    ON t.Titolo=s.TitoloTest
    AND s.MailStudente=Mail;
END$

/*-----*/
/*OPERAZIONI che riguardano SOLO DOCENTI:*/
DELIMITER $ /* 1) Inserimento di una nuova tabella di esercizio, con relativi meta-dati. */
CREATE PROCEDURE InserimentoTabellaEsercizio (
    IN NomeTabella VARCHAR(30),
    IN MailDocente VARCHAR(40))
BEGIN
    INSERT INTO TABELLA_ESERCIZIO(Nome, Creazione, NumeroRighe, MailDocente)
    VALUES (NomeTabella, now(), 0, MailDocente);
END $ DELIMITER ;

DELIMITER $ /* 2) Inserimento di una riga per una tabella di esercizio, definita dal docente. */

DELIMITER $ /* 3) Creazione di nuovo test. */
CREATE PROCEDURE CreaTest (
    IN TitoloTest VARCHAR(30),
    IN FotoTest VARCHAR(200),
    IN VisualizzaRisposteTest BOOLEAN,
    IN MailDocente VARCHAR(30)
)
BEGIN
    INSERT INTO TEST (Titolo, DataTest, Foto, VisualizzaRisposte, MailDocente)
    VALUES (TitoloTest, NOW(), FotoTest, VisualizzaRisposteTest, MailDocente);
END $ DELIMITER ;

```

```

DELIMITER $ /* 4) Creazione di un nuovo quesito con le relative risposte. */
> CREATE PROCEDURE NewSketchCodice (
    IN ProgressivoCodice INT,
    IN TitoloTest VARCHAR(30),
    IN Difficolta VARCHAR(5),
    IN DescrizioneQuesito VARCHAR(400),
    IN Soluzione VARCHAR(300)
~ )
> BEGIN
    INSERT INTO QUESITO (Progressivo, TitoloTest, Difficolta, Descrizione, NumRisposte)
    VALUES (ProgressivoCodice, TitoloTest, Difficolta, DescrizioneQuesito, 0);
    INSERT INTO SKETCH_CODICE (Progressivo, TitoloTest, Soluzione)
    VALUES (ProgressivoCodice, TitoloTest, Soluzione);
~ END $ DELIMITER ;
DELIMITER $
> CREATE PROCEDURE NewQuesitoChiuso (
    IN ProgressivoChiuso INT,
    IN TitoloTest VARCHAR(30),
    IN Difficolta VARCHAR(5),
    IN DescrizioneQuesito VARCHAR(400)
~ )
> BEGIN
    INSERT INTO QUESITO (Progressivo, TitoloTest, Difficolta, Descrizione, NumRisposte)
    VALUES (ProgressivoChiuso, TitoloTest, Difficolta, DescrizioneQuesito, 0);
    INSERT INTO QUESITO_CHIUSO (Progressivo, TitoloTest)
    VALUES (ProgressivoChiuso, TitoloTest);
~ END $ DELIMITER ;

/* 5) Abilitare / disabilitare la visualizzazione delle risposte per uno specifico test. */
DELIMITER $
> CREATE PROCEDURE VisualizzazioneRisposte (
    IN TitoloTest VARCHAR(30),
    IN VisualizzaRisposte boolean
~ )
> BEGIN
    UPDATE TEST
    SET TEST.VisualizzaRisposte = VisualizzaRisposte
    WHERE (TitoloTest = TEST.Titolo);
~ END $ DELIMITER ;

/* 6) Inserimento di un messaggio (da parte del docente). */
DELIMITER $
> CREATE PROCEDURE InserimentoMessaggioDocente (
    IN TitoloMessaggio VARCHAR(30),
    IN TestoMessaggio VARCHAR(100),
    IN TitoloTest VARCHAR(30),
    IN MailDocente VARCHAR(40))
~ )
> BEGIN
    -- Inserimento del messaggio nella tabella MESSAGGIO

    INSERT INTO MESSAGGIODOCENTE (TitoloMess, Testo, DataInserimento, TitoloTest, MailDocente)
    VALUES (TitoloMessaggio, TestoMessaggio, NOW(), TitoloTest, MailDocente);
~ END $ DELIMITER ;

```

```

/*-----*/
/*OPERAZIONI che riguardano SOLO STUDENTI*/
DELIMITER $ /* 1) Inserimento di una nuova risposta (ad un quesito di codice o un quesito chiuso). */
CREATE PROCEDURE InserimentoRisposta (IN ProgressivoQuesito INT, IN TitoloTest VARCHAR(30), IN MailStudente VARCHAR(40), IN Esito TINYINT, IN Testo VARCHAR(300))
BEGIN
    INSERT INTO risposta(ProgressivoQuesito, TitoloTest, MailStudente, Esito, Testo)
    VALUES (ProgressivoQuesito, TitoloTest, MailStudente, Esito, Testo);
END $ DELIMITER ;

DELIMITER $ /* 2) Visualizzazione dell'esito di una risposta (ad un quesito di codice o un quesito chiuso). */
CREATE PROCEDURE VisualizzaEsitoRisposta (IN ProgressivoQuesito INT, IN TitoloTest VARCHAR(30), IN MailStudente VARCHAR(30))
BEGIN
    SELECT Esito
    FROM RISPOSTA
    WHERE RISPOSTA.ProgressivoQuesito = ProgressivoQuesito
    AND RISPOSTA.MailStudente = MailStudente
    AND RISPOSTA.TitoloTest = TitoloTest;
END $ DELIMITER ;

/* 3) Inserimento di un messaggio (da parte dello studente).*/
DELIMITER $
> CREATE PROCEDURE InserimentoMessaggioStudente (
    IN TitoloMessaggio VARCHAR(30),
    IN TestoMessaggio VARCHAR(100),
    IN TitoloTest VARCHAR(30),
    IN MailStudente VARCHAR(40),
    IN MailDocente VARCHAR(40))
~
> BEGIN
    -- Inserimento del messaggio nella tabella MESSAGGIO

    INSERT INTO MESSAGGIOSTUDENTE (TitoloMess, Testo, DataInserimento, TitoloTest, MailStudente, MailDocente)
    VALUES (TitoloMessaggio, TestoMessaggio, NOW(), TitoloTest, MailStudente, MailDocente);

~
END $ DELIMITER ;

/*procedure nuove (non richieste da traccia)*/
-- aggiornamento della risposta se lo studente la cambia
DELIMITER $
CREATE PROCEDURE AggiornaRisposta(IN NuovoTesto VARCHAR(300), IN NuovoEsito TINYINT, IN Test VARCHAR(30), IN Mail VARCHAR(40), IN NuovoProgressivo INT)
> BEGIN
    UPDATE risposta SET Testo=NuovoTesto, Esito=NuovoEsito WHERE MailStudente=mail AND TitoloTest=Test AND ProgressivoQuesito=NuovoProgressivo;
~
END$
DELIMITER ;

-- visualizzazione della tabella fisica dentro il test per lo studente
DELIMITER $
CREATE PROCEDURE VisualizzaTabella (IN Test VARCHAR(30))
~
BEGIN
    SELECT DISTINCT r.NomeTabella, a.Nome AS NomeAttributo
    FROM rif_tabella_quesito as r, attributo as a
    WHERE r.NomeTabella = a.NomeTabella AND r.TitoloTest=Test;
~
END$
DELIMITER ;

-- inserimento dell'attributo nelle tabelle esercizio fisiche
DELIMITER $
CREATE PROCEDURE InserimentoAttributo (IN Tabella VARCHAR(30), IN NomeAT VARCHAR(30), IN Tipo VARCHAR(30), IN PossibileChiavePrimaria TINYINT)
~
BEGIN
    INSERT INTO Attributo (NomeTabella, Nome, Tipo, PossibileChiavePrimaria) VALUES (Tabella, NomeAT, Tipo, PossibileChiavePrimaria);
~
END $ DELIMITER ;

-- visualizzare gli attributi delle tabelle create dal docente
DELIMITER $
CREATE PROCEDURE VisualizzaAttributi (IN Tabella VARCHAR(30))
~
BEGIN
    SELECT Nome, Tipo, PossibileChiavePrimaria
    FROM attributo
    WHERE NomeTabella=Tabella;
~
END $ DELIMITER ;

```

```

-- procedura per creare la tabella fisica
DELIMITER $
CREATE PROCEDURE TabellaFisica (IN mail VARCHAR(30), IN Tabella VARCHAR(30))
BEGIN
    DECLARE fine BOOLEAN DEFAULT FALSE;
    DECLARE NomeAttributo VARCHAR(20);
    DECLARE tipoAttributo VARCHAR(50);
    DECLARE chiaveP BOOLEAN;
    DECLARE chiaviprimarie TEXT DEFAULT '';

    DECLARE cursore CURSOR FOR SELECT Nome, Tipo, PossibileChiavePrimaria FROM Attributo WHERE Attributo.NomeTabella = Tabella ORDER BY attributo.PossibileChiavePrimaria DESC;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET fine = TRUE;

    SET @sql = CONCAT('CREATE TABLE ', Tabella, ' (');

    OPEN cursore;
read_loop: LOOP
    FETCH cursore INTO NomeAttributo, tipoAttributo, chiaveP;
    IF fine THEN
        LEAVE read_loop;
    END IF;
    SET @sql = CONCAT(@sql, NomeAttributo, ' ', tipoAttributo, ', ');
    IF chiaveP THEN
        SET chiaviprimarie = CONCAT(chiaviprimarie, NomeAttributo, ', ');
    END IF;
END LOOP;
CLOSE cursore;

CLOSE cursore;

SET chiaviprimarie = SUBSTRING(chiaviprimarie, 1, LENGTH(chiaviprimarie) - 2);
SET @sql = CONCAT(@sql, 'PRIMARY KEY (', chiaviprimarie, '));');

PREPARE stmt FROM @sql;
EXECUTE stmt;
DEALLOCATE PREPARE stmt;
END $
DELIMITER ;

```

-- procedura per fare i vincoli di integrità fra tabelle

```

DELIMITER $
CREATE PROCEDURE Vincoli ( IN tabella2 VARCHAR(30), IN tabella1 VARCHAR(30), IN chiaviprimarie VARCHAR(500), IN chiaviesterne VARCHAR(500))
BEGIN
    SET @sql = CONCAT('ALTER TABLE ', tabella2, ' ADD FOREIGN KEY (', chiaviesterne, ') REFERENCES ', tabella1, '(', chiaviprimarie, ') ON DELETE CASCADE');

    select @sql;
    PREPARE stmt FROM @sql;
    EXECUTE stmt;
    DEALLOCATE PREPARE stmt;
END $
DELIMITER ;

```

-- procedura per popolare la tabella

```

DELIMITER $
CREATE PROCEDURE PopolaTabella (IN Tabella VARCHAR(30), IN valori text)
BEGIN
    DECLARE fine BOOLEAN DEFAULT FALSE;
    DECLARE NomeAttributo VARCHAR(20);

    DECLARE cursore CURSOR FOR SELECT Nome FROM Attributo WHERE Attributo.NomeTabella = Tabella ORDER BY attributo.PossibileChiavePrimaria DESC;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET fine = TRUE;

    SET @sql = CONCAT('insert into ', Tabella, ' (');

    OPEN cursore;
read_loop: LOOP
    FETCH cursore INTO NomeAttributo;
    IF fine THEN
        LEAVE read_loop;
    END IF;
    SET @sql = CONCAT(@sql, NomeAttributo, ', ');
END LOOP;
CLOSE cursore;

SET @sql = LEFT(@sql, LENGTH(@sql) - 2);
SET @sql = CONCAT(@sql, ') VALUES (');

```



```

SET @sql = LEFT(@sql, LENGTH(@sql) - 2);
SET @sql = CONCAT(@sql, ' VALUES (');

-- Aggiungi singoli apici attorno ai valori
SET @sql = CONCAT(@sql, valori, ');');

-- Debugging output (puoi rimuoverlo in produzione)
-- SELECT @sql;

PREPARE stmt FROM @sql;
EXECUTE stmt;
DEALLOCATE PREPARE stmt;

~ END $
DELIMITER ;

```

View

```
use cinciallegra;
```

```

/* STATISTICHE (Visibili da tutti gli Utenti) */
> /* 1) VISUALIZZARE LA CLASSIFICA DEGLI STUDENTI
- (SULLA BASE DEL NUMERO DI TEST COMPLETATI). */
> CREATE VIEW ClassificaConcluso(CodiceMatricola, Conteggio) AS (
    SELECT CodiceMatricola, COUNT(*) AS Conteggio
    FROM STUDENTE, SVOLGIMENTO
    WHERE (SVOLGIMENTO.MailStudente = STUDENTE.Mail) AND (SVOLGIMENTO.Stato = 'Concluso')
    GROUP BY CodiceMatricola
    ORDER BY Conteggio DESC
> /* DESC perché gli studenti vanno ordinati in ordine decrescente rispetto
- al numero di test che hanno completato (chi ha concluso più test sta in alto, chi meno sta in basso)*/
- );

/* 2) VISUALIZZARE LA CLASSIFICA DEGLI STUDENTI
(SULLA BASE DEL NUMERO DI RISPOSTE CORRETTE). */
CREATE VIEW ClassificaCorretto(CodiceMatricola, Percentuale) AS (
    SELECT CodiceMatricola, ((COUNT(CASE WHEN RISPOSTA.Esito = 'true' THEN 1 END)) / COUNT(*)) AS Percentuale
    FROM STUDENTE, RISPOSTA
    WHERE (RISPOSTA.MailStudente = STUDENTE.Mail)
    GROUP BY CodiceMatricola
    ORDER BY Percentuale DESC
);

/* 3) VISUALIZZARE LA CLASSIFICA DEI QUESITI
(IN BASE AL NUMERO DI RISPOSTE INSERITE). */
CREATE VIEW ClassificaQuesiti(Descrizione, Conteggio) AS (
    SELECT Descrizione, SUM(NumRisposte) AS Conteggio
    FROM QUESITO
    GROUP BY Descrizione
    ORDER BY Conteggio DESC
);

```

Trigger

```
/* TRIGGERS */
```

```
DELIMITER $
```

```
CREATE TRIGGER NumeroRisposte
```

```
AFTER INSERT ON Risposta
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    UPDATE quesito
```

```
    SET NumRisposte = NumRisposte + 1
```

```
    WHERE NEW.ProgressivoQuesito = quesito.Progressivo and NEW.TitoloTest=quesito.TitoloTest;
```

```
END;
```

```
$ DELIMITER ;
```

```
DELIMITER $
```

```
CREATE TRIGGER InCompletamento
```

```
AFTER INSERT ON Risposta
```

```
FOR EACH ROW
```

```
› BEGIN
```

```
    DECLARE numeroRisposte INT;
```

```
    -- Conta quante risposte sono state date dallo studente al test
```

```
    SELECT COUNT(*) INTO numeroRisposte
```

```
    FROM Risposta
```

```
    WHERE MailStudente = NEW.MailStudente AND TitoloTest = NEW.TitoloTest;
```

```
    -- Se è la prima risposta, aggiorna lo stato del test a "InCompletamento"
```

```
› IF numeroRisposte = 1 THEN
```

```
    UPDATE Svolgimento
```

```
    SET Stato = 'InCompletamento'
```

```
    WHERE svolgimento.MailStudente = NEW.MailStudente AND svolgimento.TitoloTest = NEW.TitoloTest;
```

```
~ END IF;
```

```
~ END;
```

```
$ DELIMITER ;
```

```

-- concluso
DELIMITER $
CREATE TRIGGER Concluso
AFTER INSERT ON Risposta
FOR EACH ROW
BEGIN
    DECLARE numeroRisposte INT;
    DECLARE numeroQuesiti INT;

    -- Conta quante risposte corrette sono state date dallo studente al test
    SELECT COUNT(*) INTO numeroRisposte
    FROM Risposta
    WHERE MailStudente = NEW.MailStudente AND TitoloTest = NEW.TitoloTest AND Esito=1;

    -- Conta i quesiti totali del test
    SELECT COUNT(*) INTO numeroQuesiti
    FROM quesito
    WHERE TitoloTest = NEW.TitoloTest;

    -- Se è la prima risposta, aggiorna lo stato del test a "InCompletamento"
    IF numeroRisposte = numeroQuesiti THEN
        UPDATE Svolgimento
        SET Stato = 'Concluso'
        WHERE svolgimento.MailStudente = NEW.MailStudente AND svolgimento.TitoloTest = NEW.TitoloTest;
    END IF;
END;
$ DELIMITER ;

```

```

-- SE SI FA UN AGGIORNAMENTO IN RISPOSTA
DELIMITER $
CREATE TRIGGER Concluso2
AFTER UPDATE ON Risposta
FOR EACH ROW
BEGIN
    DECLARE numeroRisposte INT;
    DECLARE numeroQuesiti INT;

    -- Conta quante risposte corrette sono state date dallo studente al test
    SELECT COUNT(*) INTO numeroRisposte
    FROM Risposta
    WHERE MailStudente = NEW.MailStudente AND TitoloTest = NEW.TitoloTest AND Esito=1;

    -- Conta i quesiti totali del test
    SELECT COUNT(*) INTO numeroQuesiti
    FROM quesito
    WHERE TitoloTest = NEW.TitoloTest;

    -- Se è la prima risposta, aggiorna lo stato del test a "InCompletamento"
    IF numeroRisposte = numeroQuesiti THEN
        UPDATE Svolgimento
        SET Stato = 'Concluso'
        WHERE svolgimento.MailStudente = NEW.MailStudente AND svolgimento.TitoloTest = NEW.TitoloTest;
    END IF;
END;
$ DELIMITER ;

-- Se il docente imposta VisualizzaRisposte = true
DELIMITER $
CREATE TRIGGER ConclusoVR
AFTER UPDATE ON Test
FOR EACH ROW
BEGIN
    IF NEW.VisualizzaRisposte = 1 THEN
        UPDATE Svolgimento
        SET Stato = 'Concluso'
        WHERE Svolgimento.TitoloTest = NEW.Titolo;
    END IF;
END;
$ DELIMITER ;

```