Alma Mater Studiorum - Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica per il Management

Piattaforma ESQL

DOCUMENTAZIONE SVOLTA DA:
Canghiari Matteo
De Rosa Davide
Nadifi Ossama

Anno Accademico 2023/2024

1 Analisi dei requisiti

All'interno di questa prima sezione, si adotta un approccio orientato ad un'analisi degli aspetti principali inerenti al progetto, mediante una serie di azioni mirate per rendere il più comprensibile possibile il documento di specifica, attraverso la scelta del corretto livello di astrazione, la standardizzazione della struttura delle frasi oppure tramite la decomposizione del testo in espressioni omogenee.

1.2 Documento di specifica

Tutti gli utenti della piattaforma dispongono di un indirizzo email, nome, cognome e, opzionalmente, di un recapito telefonico. Gli utenti possono essere suddivisi in due categorie principali: docenti e studenti. I docenti forniscono informazioni sul dipartimento di afferenza e sul corso di cui sono titolari. Gli studenti forniscono informazioni sull'anno di immatricolazione e un codice alfanumerico univoco. I docenti hanno la possibilità di creare tabelle di esercizio, ognuna caratterizzata da un nome, una data di creazione e un numero di righe specificato. Le tabelle di esercizio sono correlate a un insieme di attributi, ciascuno con un nome, un tipo e la possibilità di far parte della chiave primaria della tabella di esercizio. Inoltre, i docenti possono creare test, ciascuno con un titolo univoco, una data di creazione e la possibilità di includere una foto. Ogni test può contenere diversi quesiti, ciascuno con un numero progressivo, un livello di difficoltà, un campo descrizione e un numero di risposte. I quesiti fanno riferimento a una o più tabelle di esercizio creati dal docente. I quesiti possono appartenere esclusivamente a due categorie: quesiti a domanda chiusa e quesiti di codice. Le domande chiuse hanno una serie di opzioni di risposta, ciascuna con una numerazione e un campo testo. I quesiti di codice hanno una o più soluzioni definite come sketch di codice. Ogni test ha un campo booleano VisualizzaRisposte, che, se impostato su true, rende visibili le risposte dei quesiti agli studenti; altrimenti, rimangono nascoste. Gli studenti possono svolgere un test, fornendo una o più risposte per ciascun quesito. Si tiene traccia del completamento del test, ovvero la data di inserimento della prima risposta, la data di inserimento dell'ultima risposta e lo stato. Nel caso di quesiti a domanda chiusa, la risposta consiste potenzialmente nell'insieme dell'opzioni disponibili. Nel caso di quesiti di codice, la risposta consiste in un campo testo. È prevista la possibilità per gli studenti di inviare più risposte per lo stesso quesito in istanti diversi. Ogni risposta dispone di un campo esito, un campo booleano che definisce la correttezza della risposta fornita, sia che si tratti di una domanda chiusa sia che si tratti di un quesito di codice. È anche possibile inviare messaggi. Ogni messaggio ha un titolo, un campo testo, una data di inserimento e fa riferimento ad uno specifico test. Il messaggio può essere inviato da un docente o da uno studente. Nel primo caso, i destinatari saranno gli studenti; nel secondo caso, il destinatario sarà il determinato docente creatore del test.

1.3 Decomposizione in gruppi di frasi

Di seguito sono descritti i concetti essenziali raggruppati sulla base di medesime caratteristiche, affinchè sia definito un supporto concreto per successive fasi di sviluppo, costituito da:

- **UTENTE**

Tutti gli utenti dispongono di: email, nome, cognome e di un possibile recapito telefonico. Gli utenti sono suddivisi in due tipologie: docenti e studenti.

- **STUDENTE**

Gli studenti dispongono di un campo anno di immatricolazione e di un codice alfanumerico. Gli studenti possono svolgere un test, inserendo una o più risposte per ciascun quesito.

- **DOCENTI**

I docenti dispongono del nome del dipartimento di afferenza e nome del corso di cui sono titolari. I docenti possono creare delle tabelle di esercizio. Devono essere inseriti dai docenti anche i vincoli di integrità referenziale tra i differenti attributi delle tabelle di esercizio. In aggiunta ogni docente può creare dei test.

- **TABELLE_ESERCIZIO**

Ogni tabella di esercizio dispone di nome, data di creazione e un numero di righe specificato. Inoltre, ogni tabella di esercizio dispone di un insieme di attributi.

- **ATTRIBUTO**

Ogni attributo dispone di un nome, un tipo e può essere parte della chiave primaria della tabella di esercizio.

- **TEST**

Ogni test dispone di un titolo univoco, una data di creazione e di una possibile foto. Ogni test include una serie di quesiti. Ogni test ha un campo booleano VisualizzaRisposte, che, se impostato su true, rende visibili le risposte dei quesiti agli studenti; altrimenti, rimangono nascoste.

- **QUESITO**

Ogni quesito dispone di un numero progressivo univoco, ma solo all'interno di un test, un livello di difficoltà, un campo descrizione e un numero di risposte. I quesiti fanno riferimento ad una o più tabelle di esercizio create dal docente. I quesiti sono esclusivamente di due categorie: domande a risposta chiusa oppure quesiti di codice.

- **DOMANDA_CHIUSA**

La domanda chiusa dispone di una serie di opzioni di risposta. Nel caso di quesiti a domanda chiusa, la risposta consiste potenzialmente nell'insieme dell'opzioni disponibili.

- **OPZIONI_RISPOSTA**

Ogni opzione dispone di una numerazione, univoca rispetto ad uno specifico quesito, ed un campo di testo.

- **DOMANDA_CODICE**

Il quesito di codice dispone di una o più soluzioni. Nel caso di quesiti di codice, la risposta consiste in un campo di testo.

- **SKETCH_CODICE**

Gli sketch di codice in SQL implementano query che restituiscano quanto richiesto dal quesito.

- **COMPLETAMENTO**

Si vuole tenere traccia del completamento del test, ossia: data di inserimento della prima risposta, data di inserimento dell'ultima risposta, stato.

- **RISPOSTA**

  Ogni risposta dispone di un campo di esito, che può valere true o false a seconda che la risposta fornita dallo studente coincida con l'opzione del quesito a domanda chiusa oppure che la risposta produca l'output desiderato nel caso di quesiti di codice.

- **MESSAGGI**

  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 oppure da uno studente. Nel primo caso, i destinatari saranno tutti gli studenti; nel secondo caso, il destinatario sarà il determinato docente.

1.4 Lista delle operazioni

Come da titolo, sono riportate l'insieme delle possibili operazioni sui dati individuate durante l'analisi del documento di specifica, costituito da:

OPERAZIONE 1. Inserire un nuovo utente

OPERAZIONE 2. Visualizzare i dati degli studenti

OPERAZIONE 3. Registrare un nuovo profilo utente alla piattaforma

OPERAZIONE 4. Autenticare l'accesso di un profilo utente alla piattaforma

OPERAZIONE 5. Inserire nuovi quesiti

OPERAZIONE 6. Inserire una nuova tabella di esercizio, con i propri meta-dati

OPERAZIONE 7. Inserire nuove opzioni di risposta

OPERAZIONE 8. Visualizzare tutti i quesiti associati a differenti test

OPERAZIONE 9. Inserire una o più risposte rispetto ad un certo quesito

OPERAZIONE 10. Visualizzare l'esito della risposta inserita da uno studente

OPERAZIONE 11. Modificare la modalità di visualizzazione delle risposte

OPERAZIONE 12. Inserire un nuovo messaggio

OPERAZIONE 13. Visualizzare le conversazioni effettuate

1.5 Tavola media dei volumi

In questa sezione è specificato il numero stimato di istanze per ogni entità e relazione dello schema. I valori sono necessariamente approssimati, ma oltre tutto indicativi. Si prende come riferimento una realtà universitaria.

Concetto	**Tipo**	**Volume**
Utente	Entità	305
Studente	Entità	300
Docente	Entità	5
Tabella_Esercizio	Entità	50
Attributo	Entità	200
Test	Entità	10
Quesito	Entità	100
Domanda_Chiusa	Entità	50
Opzione_Risposta	Entità	150
Domanda_Codice	Entità	50
Sketch_Codice	Entità	50
Messaggio	Entità	610
Messaggio_Studente	Entità	600
Creazione	Relazione	50
Completamento	Relazione	3000
Invio	Relazione	18000
Ricezione	Relazione	10
Pubblicazione	Relazione	6100
Risposta	Relazione	30000
Composizione	Relazione	1000
Afferenza	Relazione	5000
Combinazione	Relazione	10000
Vincolo Integrità	Relazione	40000
Disposizione	Relazione	7500
Soluzione	Relazione	2500

Table 1: Stima della tavola media dei volumi riferita al progetto svolto.

1.6 Glossario dei termini

Grazie alla sezione riferita alla decomposizione delle frasi secondo caratteristiche comuni, è possibile realizzare un glossario dei termini, capace di favorire una panoramica delle nozioni principali. Il glossario, rispetto a quanto svolto, si compone di:

Termine	**Descrizione**	**Sinonimi**	**Collegamenti**
Utente	Persona utilizzatrice della piattaforma ESQL	.	Docente, Studente
Docente	Docente titolare del corso. Somministra dei test, crea tabelle di esercizio e invia messaggi agli studenti	.	Tabella_Esercizio, Test, Messaggio
Studente	Studente dei corsi. Può svolgere più prove, oltre a rispondere più volte allo stesso quesito	.	Test, Quesito, Messaggio
Tabella_Esercizio	Tabella di esercizio contenente i metadati necessari per la realizzazione di test	.	Docente, Attributo, Quesito
Attributo	Attributo delle tabelle di esercizio	.	Tabella_Esercizio
Test	Test ideati dai docenti e somministrati agli studenti, include un insieme di quesiti	.	Docente, Studente, Quesito, Tabella_Esercizio, Messaggio
Quesito	Quesito sottoposto agli studenti del corso, può assumere una singola tipologia tra domanda chiusa o quesito di codice	.	Studente, Test, Tabella_Esercizio, Domanda_Chiusa, Domanda_Codice
Domanda_Chiusa	Domanda a risposta chiusa, inerente ad un quesito posto agli studenti, possiede più di un'opzione di risposta	Risposta chiusa	Quesito, Opzione_Risposta
Opzione_Risposta	Opzioni di risposta riferite ad uno specifico quesito	.	Domanda_Chiusa
Domanda_Codice	Quesito di codice SQL, per la costruzione di query che restituiscano il risultato voluto	Quesito di codice	Quesito, Sketch_Codice
Sketch_Codice	Sketch risolutivi rispetto al quesito di codice posto, quindi può esistere più di una soluzione	Opzione risposta del codice	Domanda_Codice
Completamento	Stato di completamento dei test da parte degli studenti	.	Studente, Test
Risposta	Risposta formulata da uno studente per la risoluzione dei quesiti somministrati	.	Studente, Quesito
Messaggio	Comunicazioni inviate e ricevute tra docenti e studenti, una comunicazione è riferita ad un solo docente e a tutti gli studenti dello specifico corso	Comunicazione	Studente, Docente, Test

Table 2: Glossario dei termini individuati all'interno del documento di specifica.

2 Progettazione concettuale

Definito il primo step, inerente all'analisi dei concetti e termini di maggior spessore, il passo successivo comprende la modellazione dello schema E-R. Tale diagramma è adottato per la rappresentazione concettuale dei dati ad alto livello di astrazione, crocevia essenziale per la realizzazione di un qualsiasi database. Si compone non solo della rappresentazione grafica, ma anche di strumenti descrittivi, dedicati a tutte quelle caratteristiche non riproducibili attraverso lo schema.

2.1 Modello E-R

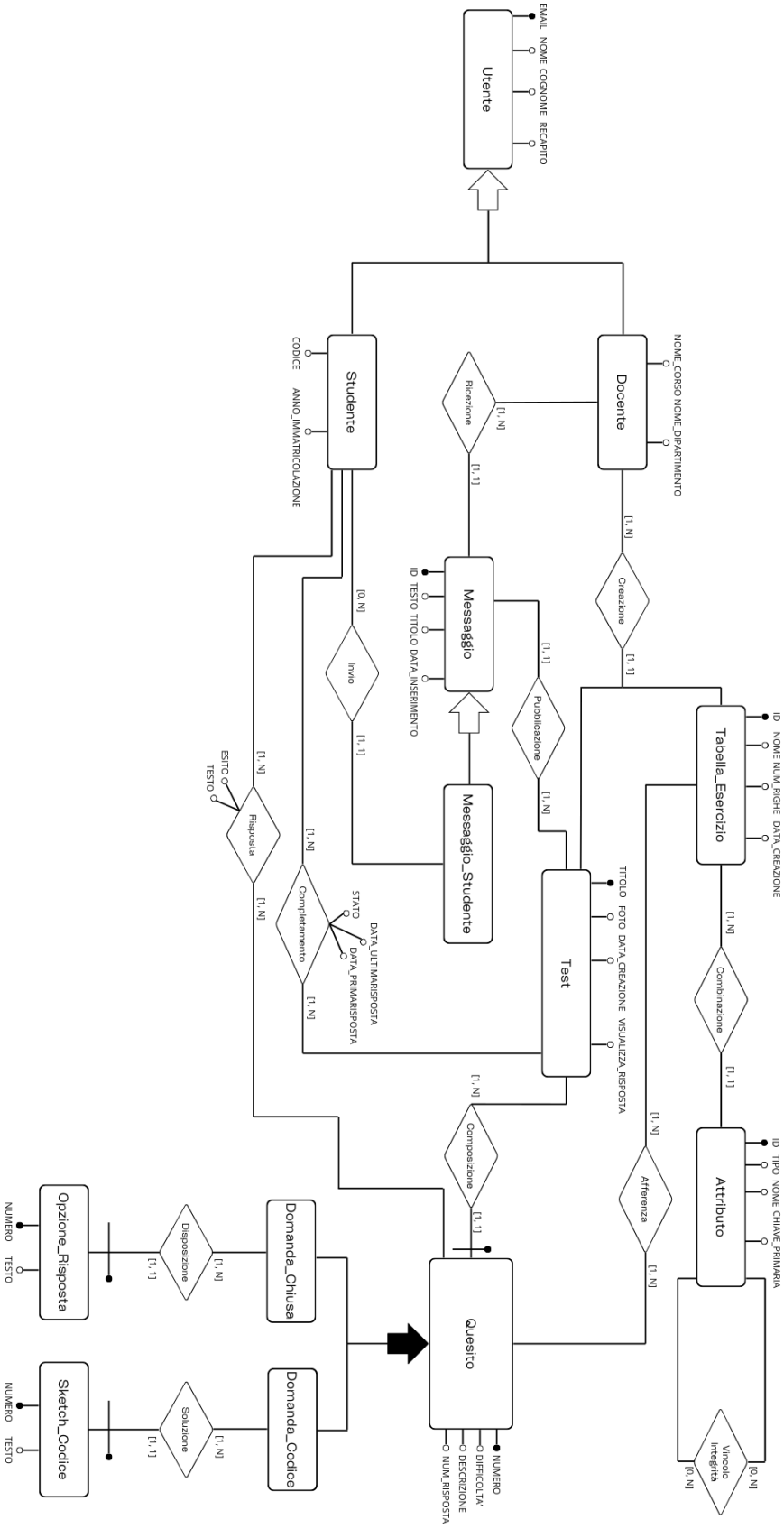

Figure 1: Modello E-R precedente al raffinamento.

2.2 Dizionario delle entità

Entità	Descrizione	Attributi	Identificatore
Utente	Utilizzatore generale dell'applicativo	Email, Password, Nome, Cognome, Telefono	Email
Studente	Studente fruitore della piattaforma per la risoluzione dei quesiti posti	Email_Studente, Anno_Immatricolazione, Codice	Email_Studente
Docente	Docente creatore e ideatore di quesiti e tabelle di esercizio	Email_Docente, Nome_Dipartimento, Nome_Corso	Email_Docente
Tabella_Esercizio	Tabelle contenenti i metadati per la realizzazione di eventuali quesiti	Id, Email_Docente, Nome, Data_Creazione, Num_Righe	Id
Attributo	Attributi parte costituente delle tabelle di esercizio, finalizzati per la realizzazione di quesiti	Id, Id_Tabella, Tipo, Nome, Chiave_Primaria	Id
Test	Test indica l'insieme di quesiti svolti dagli studenti e creati dal docente	Titolo, Email_Docente, Foto, Data_Creazione, Visualizza_Risposte	Titolo
Quesito	Quesito relativo a tematiche svolte durante il corso	Id, Titolo_Test, Difficoltà, Num_Risposte, Descrizione	Id, Titolo_Test
Domanda_Chiusa	Tipologia di quesito, rappresentante una domanda a scelta multipla	Id_Domanda_Chiusa, Titolo_Test	Id_Domanda_Chiusa, Titolo_Test
Opzione_Risposta	Opzioni di risposta relative ad una domanda chiusa	Id, Id_Domanda_Chiusa, Titolo_Test, Testo, Soluzione	Id, Id_Domanda_Chiusa, Titolo_Test
Domanda_Codice	Tipologia di quesito, richiedente la formulazione di query SQL	Id_Domanda_Codice, Titolo_Test	Id_Domanda_Codice, Titolo_Test
Skecth_Codice	Skecth di codice SQL risulutivi rispetto al quesito somministrato	Id, Id_Domanda_Codice, Titolo_Test, Testo, Soluzione	Id, Id_Domanda_Codice, Titolo_Test
Messaggio	Comunicazione inviata dal docente a tutti gli studenti	Id, Email_Docente, Titolo_Test, Testo, Titolo, Data_Inserimento	Id
Messaggio_Studente	Messaggio inviato dallo studente al docente del corso	Id_Messaggio_Studente, Email_Studente	Id_Messaggio_Studente

Table 3: Descrizione dell'entità del modello E-R.

2.3 Dizionario delle relazioni

Relazione	Descrizione	Componenti	Attributi
Creazione	Creazione da parte di docenti di tabelle di esercizio e di test	Docente, Tabella_Esercizio, Test	.
Completamento	Completamento di un test somministrato da parte degli studenti	Studente, Test	Email_Studente, Titolo_Test, Stato, Data_Ultima_Risposta, Data_Prima_Risposta
Invio	Invio di messaggi da parte di studenti	Studente, Messaggio_Studente	.
Ricezione	Invio e ricezione di comunicazioni dal docente del corso	Docente, Messaggio	.
Pubblicazione	Pubblicazione di comunicazioni afferenti ad uno specifico test	Messaggio, Test	.
Risposta	Risposta formulata dagli studenti in relazione ad uno specifico quesito	Studente, Quesito	Email_Studente, Id_Quesito, Titolo_Test, Testo, Esito
Composizione	Composizione di un insieme di quesiti rispetto ad un determinato test	Quesito, Test	.
Afferenza	Afferenza dei quesiti ideati relativamente a tabelle di esercizio	Quesito, Tabella_Esercizio	.
Combinazione	Combinazione di attributi per la costruzione di tabelle di esercizio	Attributo, Tabella_Esercizio	.
Vincolo Integrità	Vincolo di integrità che mantiene tutti i vincoli referenziali tra attributi di tabelle di esercizio	Attributo	Referente, Referenziato
Disposizione	Disposizione del numero complessivo di opzioni di risposta relative alla domanda chiusa sottoposta	Opzione_Risposta, Domanda_Chiusa	.
Soluzione	Soluzione insieme delle query SQL che risolvono la domanda di codice	Sketch_Codice, Domanda_Codice	.

Table 4: Descrizione delle relazioni del modello E-R.

2.4 Tavola delle business rules

Regole di vincolo
Il campo Codice alfanumerico degli studenti deve avere una lunghezza pari a 16 caratteri
Il docente può inserire i vincoli di integrità referenziale tra gli attributi che compongano tabelle di esercizio
Il numero progressivo associato ad un quesito è univoco, ma solo all'interno di uno specifico test
Il numero delle opzioni di risposta, sia per Domande_Chiuse che per Domande_Codice, è univoco, ma solo all'interno dello specifico quesito
Uno studente può sottomettere più risposte per lo stesso quesito, ma solo se il test non è in stato Concluso
Un messaggio inviato da un docente è recapitato da tutti gli studenti del corso, invece un messaggio comunicato da uno studente è ricevuto dallo specifico docente
Regole di derivazione
Il livello di difficoltà di ogni test consiste in un campo enum, che può assumere esclusivamente tre valori: Basso, Medio oppure Difficile
Il campo Num_Risposte nell'entità Quesito è una ridondanza concettuale
L'attributo Visualizza_Risposte è un campo booleano, permette di visualizzare o meno le risposte dei quesiti
I campi Data_PrimaRisposta e Data_UltimaRisposta, della relazione Completamento, devono essere espressi su scala temporale
L'attributo della relazione Completamento è un campo enum, il quale può assumere esclusivamente tre valori: Aperto, InCompletamento e Concluso
Il campo Esito della relazione Risposta è un attributo booleano, definisce la correttezza della risposta sottomessa

Table 5: Descrizione delle regole di vincolo e di derivazione non attuabili tramite il modello concettuale

3 Progettazione logica

L'obiettivo di tale sezione promuove la realizzazione del modello logico a partire dalle informazioni del modello E-R. Tuttavia è bene attuare un insieme di possibili passaggi che possano favorire la traduzione, adeguando tematiche di efficienza e correttezza. Pertanto, come da capitolo seguente, è definita la ristrutturazione del diagramma E-R, affinchè sia agevolata la traduzione secondo il modello logico, ottimizzando il processo nella sua interezza.

3.1 Modello E-R raffinato

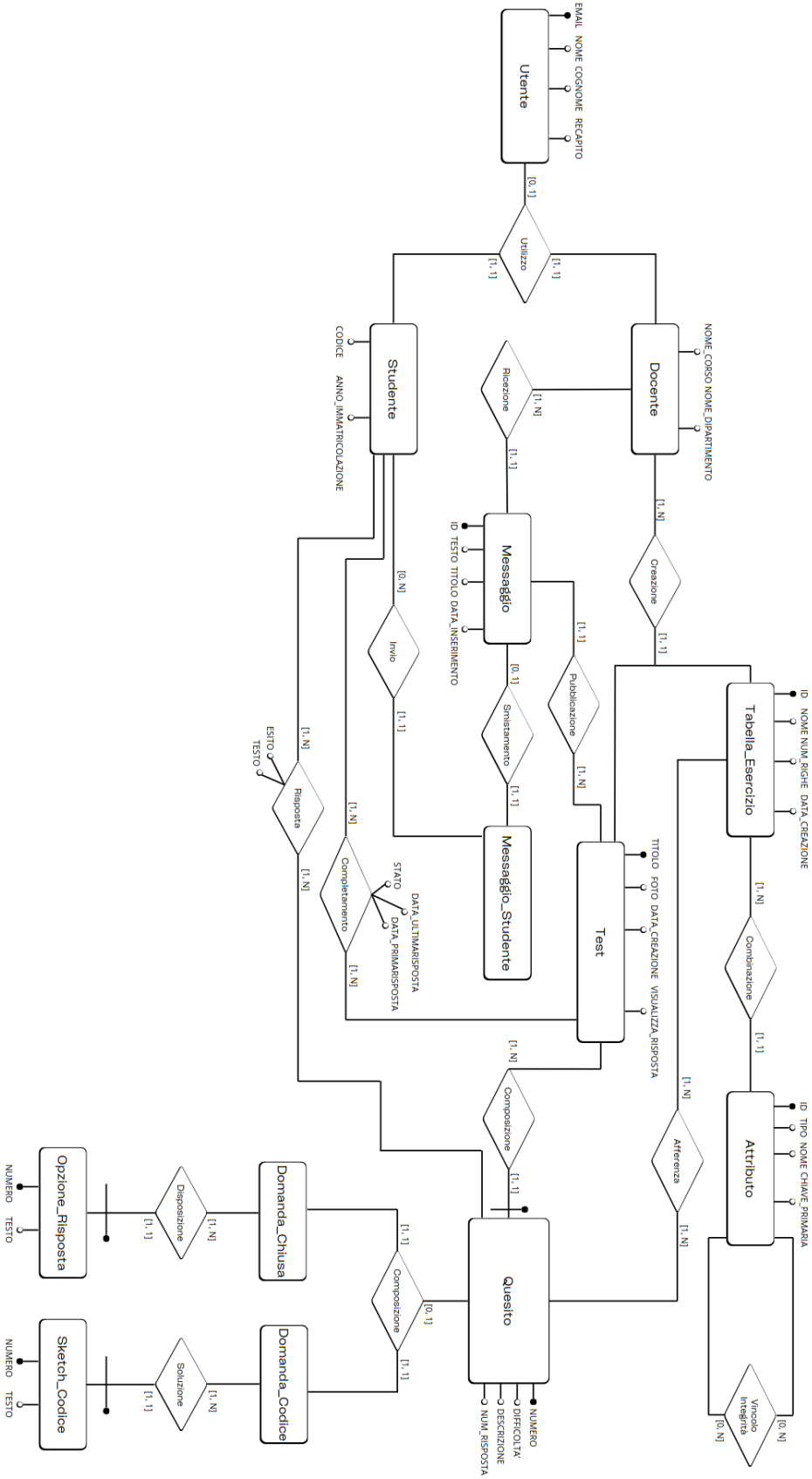

Figure 2: Modello E-R successivo al raffinamento.

3.2 Analisi delle ridondanze

In questa sezione sono riportate tutte le considerazioni necessarie per comprendere al meglio l'impatto delle ridondanze sullo schema E-R ideato; con conseguente definizione del costo operazionale sui dati di riferimento. Una fase simile è cruciale per stabilire se determinate informazioni comportano svantaggi significativi, come ad esempio maggiore occupazione di memoria oppure maggiore complessità degli aggiornamenti.

Si definiscono le seguenti annotazioni:

- S_r, lo schema E-R **con ridondanza**
- S, lo schema E-R **senza ridondanza**

Sono specificate un totale di quattro operazioni, relative al campo *Num_Risposte* dell'entità *Quesito*, le quali si contraddistinguono in:

OP1
Aggiungere una nuova risposta ad un quesito esistente (10 volte/mese, interattiva)

OP2
Rimuovere un quesito e tutte le risposte ottenute (2 volte/mese, batch)

OP3
Visualizzare tutti gli utenti presenti nella piattaforma (1 volte/mese, batch)

OP4
Contare il numero di risposte per ciascun quesito presente nella piattaforma (2 volte/mese, interattiva)

Successivamente si calcolano i costi operazionali dello schema E-R con ridondanza, esplicitando tutti gli elementi necessari, suddivisi in:

TAVOLA ACCESSI.
1 per RISPOSTA, 1 per QUESITO
C(OP1) = 10 * 1 * (2 * 2 + 0) = 40

TAVOLA ACCESSI.
10 per RISPOSTA, 1 per QUESITO, 1 per DOMANDA_CHIUSA o 1 per DOMANDA_CODICE
C(OP2) = 2 * 0.5 * (2 * 12 + 0) = 24

TAVOLA ACCESSI.
50 per UTENTE, 50 per STUDENTE o 50 per DOCENTI
C(OP3) = 1 * 0.5 * (0 + 100) = 50

TAVOLA ACCESSI.
20 per QUESITO, 20 per DOMANDA_CHIUSA o 20 DOMANDA_CODICE
C(OP4) = 2 * 1 * (0 + 40) = 80

$C(S_r) = 40 + 24 + 50 + 80 = 194$

Si calcolano i costi operazionali dello schema E-R senza ridondanza, suddivisi in:

TAVOLA ACCESSI.
1 per RISPOSTA
C(OP1) = 10 * 1 * (2 * 1 + 0) = 20

TAVOLA ACCESSI.
10 per RISPOSTA, 1 per QUESITO, 1 per DOMANDA_CHIUSA o 1 per DOMANDA_CODICE
C(OP2) = 2 * 0.5 * (2 * 12 + 0) = 24

TAVOLA ACCESSI.
50 per UTENTE, 50 per STUDENTE o 50 per DOCENTI
C(OP3) = 1 * 0.5 * (0 + 100) = 50

TAVOLA ACCESSI.
200 per RISPOSTA, 20 per QUESITO, 20 per DOMANDA_CHIUSA o 20 per DOMANDA_CODICE
C(OP4) = 2 * 1 * (0 + 240) = 480

C(S) = 20 + 24 + 50 + 480 = 564

Si osserva ora l'occupazione di memoria di entrambi i diagrammi, in cui si manifesta:

- M(S_r) = X + 20 * 4 = X + 80 byte
- M(S) = X byte

Terminate tutte le operazioni riferite all'analisi delle ridondanze, si osserva come il mantenimento comporti a vantaggi significativi, piuttosto che l'eliminazione. Infatti il rapporto $C(S) \div C(S_r)$ restituisce un valore che si aggira attorno a 3, in questo caso è conveniente mantenere l'attributo *Num_Risposte*, dettata anche dall'irrisorio overhead introdotto, pari a 80 byte.

3.3 Lista delle tabelle con i vincoli di chiave

Dopo aver svolto le fasi principali che agevolano il processo di traduzione, nella sezione consecutiva si osserva la descrizione del modello logico di riferimento, il quale fornisce l'insieme di tabelle che caratterizzano il database e i differenti attributi relativi ad ogni specifica relazione. Si evidenzia la presenza delle chiavi primarie, segnalate attraverso una sottolineatura delle colonne che compongano il vincolo.

Utente(<u>EMAIL</u>, PASSWORD, NOME, COGNOME, TELEFONO)
Studente(<u>EMAIL_STUDENTE</u>, ANNO_IMMATRICOLAZIONE, CODICE)
Docente(<u>EMAIL_DOCENTE</u>, NOME_DIPARTIMENTO, NOME_CORSO)
Tabella_Esercizio(<u>ID</u>, EMAIL_DOCENTE, NOME, DATA_CREAZIONE, NUM_RIGHE)
Attributo(<u>ID</u>, ID_TABELLA, TIPO, NOME, CHIAVE_PRIMARIA)
Vincolo_integrita(<u>REFERENTE</u>, <u>REFERENZIATO</u>)
Test(<u>TITOLO</u>, EMAIL_DOCENTE, FOTO, DATA_CREAZIONE, VISUALIZZA_RISPOSTA)
Quesito(<u>ID</u>, <u>TITOLO_TEST</u>, DIFFICOLTA, NUM_RISPOSTE, DESCRIZIONE)
Afferenza(<u>ID_QUESITO</u>, <u>TITOLO_TEST</u>, <u>ID_TABELLA</u>)

Domanda_Chiusa(ID_DOMANDA_CHIUSA, TITOLO_TEST)

Opzione_Risposta(ID, ID_DOMANDA_CHIUSA, TITOLO_TEST, TESTO, SOLUZIONE)

Domanda_Codice(ID_DOMANDA_CODICE, TITOLO_TEST)

Skecth_Codice(ID, ID_DOMANDA_CODICE, TITOLO_TEST, TESTO, SOLUZIONE)

Completamento(TITOLO_TEST, EMAIL_STUDENTE, STATO, DATA_ULTIMA_RISPOSTA, DATA_PRIMA_RISPOSTA)

Risposta(EMAIL_STUDENTE, ID_QUESITO, TITOLO_TEST, TESTO, ESITO)

Messaggio(ID, EMAIL_DOCENTE, TITOLO_TEST, TESTO, TITOLO, DATA_INSERIMENTO)

Messaggio_Studente(ID_MESSAGGIO_STUDENTE, EMAIL_STUDENTE)

3.4 Lista dei vincoli inter-relazionali

Quest'ultima parte del capitolo, riporta in maniera esplicita tutti i vincoli inter-relazionali che intercorrono tra le differenti tabelle, disposti nello stesso ordine in cui sono visualizzate le relazioni nel paragrafo precedente.

Studente.EMAIL_STUDENTE –> Utente.EMAIL

Docente.EMAIL_DOCENTE –> Utente.EMAIL

Tabella_Esercizio.EMAIL_DOCENTE –> Docente.EMAIL_DOCENTE

Attributo.ID_TABELLA –> Tabella_Esercizio.ID

Vincolo_Integrità.REFERENTE –> Attributo.ID

Vincolo_Integrità.REFERENZIATO –> Attributo.ID

Test.EMAIL_DOCENTE –> Docente.EMAIL_DOCENTE

Quesito.TITOLO_TEST –> Test.TITOLO

Afferenza.ID_QUESITO –> Quesito.ID

Afferenza.TITOLO_TEST –> Quesito.TITOLO_TEST

Afferenza.ID_TABELLA –> Tabella_Esercizio.ID

Domanda_Chiusa.ID_DOMANDA_CHIUSA –> Quesito.ID

Domanda_Chiusa.TITOLO_TEST –> Quesito.TITOLO_TEST

Opzione_Risposta.ID_DOMANDA_CHIUSA –> Domanda_Chiusa.ID_DOMANDA_CHIUSA

Opzione_Risposta.TITOLO_TEST –> Domanda_Chiusa.TITOLO_TEST

Domanda_Codice.ID_DOMANDA_CODICE –> Quesito.ID

Domanda_Codice.TITOLO_TEST –> Quesito.TITOLO_TEST

Skecth_Codice.ID_DOMANDA_CODICE –> Domanda_Codice.ID_DOMANDA_CODICE

Skecth_Codice.TITOLO_TEST –> Domanda_Codice.TITOLO_TEST

Completamento.EMAIL_STUDENTE –> Studente.EMAIL_STUDENTE

Completamento.TITOLO_TEST –> Test.TITOLO

Risposta.EMAIL_STUDENTE –> Studente.EMAIL_STUDENTE

Risposta.ID_QUESITO –> Quesito.ID

Risposta.TITOLO_TEST –> Quesito.TITOLO_TEST

Messaggio.EMAIL_DOCENTE –> Docente.EMAIL_DOCENTE

Messaggio.TITOLO_TEST –> Test.TITOLO

Messaggio_Studente.ID_MESSAGGIO_STUDENTE –> Messaggio.ID

Messaggio_Studente.TITOLO_TEST –> Studente.EMAIL_STUDENTE

4 Descrizione delle funzionalità

Registrazione
Studenti e docenti hanno l'opportunità di registrarsi alla piattaforma, affinchè possano usufruire delle funzionalità implementate.

Accedi
Nuovamente, studenti e docenti, dopo essersi registrati, devono accedere alla piattaforma pur di poter utilizzare le funzionalità implementate.

Logout
L'utente qualora abbia terminato di utilizzare la piattaforma, detiene la possibilità di disconnetersi, visualizzando la schermata home.

Creazione
In relazione alla funzionalità proposta, occorre suddividere le azioni che uno studente oppure un docente potrebbe eseguire. Pertanto di seguito è definita una breve suddivisione, in cui:

- Docente, effettuato l'accesso può creare una nuova tabella, un nuovo test, un nuovo quesito ed infine inviare messaggi a tutti gli studenti
- Studente, effettuato l'accesso può inviare messaggi allo specifico docente titolare del test

Eliminazione
L'eliminazione riguarda solo la figura del docente, pertanto esso è in grado di eliminare una tabella, se non possiede vincoli di integrità referenziale, un quesito e un test.

Svolgi test
In questo caso si tratta di una funzionalità circoscritta agli studenti, per cui, visualizzando i test disponibili, uno studente ha la possibilità di svolgere un tentativo risolutivo del test di riferimento; ciò può avvennire più volte anche per lo stesso test.

Visualizza
Come avviene per la creazione, occorre distinguere le azioni che possano essere conseguite a seconda della tipologia dell'utente, suddivise in:

- Docente, può visualizzare le specifiche di una tabella, di un quesito, di un test, dei messaggi inviati e ricevuti
- Studente, può visualizzare le risposte, le soluzioni e il file multimediale connessi ad un test, i messaggi inviati e ricevuti

5 SQL

Codice SQL completo dello schema della base di dati.

```sql
DROP DATABASE IF EXISTS ESQLDB;
CREATE DATABASE IF NOT EXISTS ESQLDB;

USE ESQLDB;

CREATE TABLE Utente(
    EMAIL VARCHAR(255) PRIMARY KEY,
    PSWD VARCHAR(255),
    NOME VARCHAR(255) NOT NULL,
    COGNOME VARCHAR(255) NOT NULL,
    TELEFONO INT(10)
)ENGINE=INNODB ;

CREATE TABLE Studente(
    EMAIL_STUDENTE VARCHAR(255) PRIMARY KEY,
    ANNO_IMMATRICOLAZIONE INT(4) NOT NULL,
    CODICE VARCHAR(16) NOT NULL,
    FOREIGN KEY (EMAIL_STUDENTE) REFERENCES Utente(EMAIL) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Docente(
    EMAIL_DOCENTE VARCHAR(255) PRIMARY KEY,
    NOME_DIPARTIMENTO VARCHAR(255) NOT NULL,
    NOME_CORSO VARCHAR(255) NOT NULL,
    FOREIGN KEY (EMAIL_DOCENTE) REFERENCES Utente(EMAIL) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Tabella_Esercizio(
    ID INT AUTO_INCREMENT PRIMARY KEY,
    NOME VARCHAR(255) NOT NULL,
    DATA_CREAZIONE DATETIME NOT NULL,
    NUM_RIGHE INT NOT NULL,
    EMAIL_DOCENTE VARCHAR(255) NOT NULL,
    FOREIGN KEY(EMAIL_DOCENTE) REFERENCES Docente(EMAIL_DOCENTE) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Attributo(
    ID INT AUTO_INCREMENT PRIMARY KEY,
    ID_TABELLA INT,
    TIPO VARCHAR(255) NOT NULL,
    NOME VARCHAR(255) NOT NULL,
    CHIAVE_PRIMARIA BOOLEAN NOT NULL,
    FOREIGN KEY(ID_TABELLA) REFERENCES Tabella_Esercizio(ID) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Vincolo_Integrita(
    REFERENTE INT,
    REFERENZIATO INT,
    PRIMARY KEY(REFERENTE, REFERENZIATO),
    FOREIGN KEY(REFERENTE) REFERENCES Attributo(ID) ON DELETE CASCADE,
    FOREIGN KEY(REFERENZIATO) REFERENCES Attributo(ID) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Test(
    TITOLO VARCHAR(255) PRIMARY KEY,
```

```
    EMAIL_DOCENTE VARCHAR(255) NOT NULL,
    FOTO LONGBLOB,
    DATA_CREAZIONE DATE NOT NULL,
    VISUALIZZA_RISPOSTE BOOLEAN NOT NULL,
    FOREIGN KEY (EMAIL_DOCENTE) REFERENCES Docente(EMAIL_DOCENTE) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Quesito(
    ID INT,
    TITOLO_TEST VARCHAR(255),
    DIFFICOLTA ENUM('BASSO', 'MEDIO', 'ALTO') NOT NULL,
    NUM_RISPOSTE INT NOT NULL,
    DESCRIZIONE VARCHAR(255) NOT NULL,
    PRIMARY KEY(ID, TITOLO_TEST),
    FOREIGN KEY(TITOLO_TEST) REFERENCES Test(TITOLO) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Afferenza(
    ID_QUESITO INT,
    TITOLO_TEST VARCHAR(255),
    ID_TABELLA INT,
    PRIMARY KEY(ID_QUESITO, TITOLO_TEST, ID_TABELLA),
    FOREIGN KEY(ID_QUESITO, TITOLO_TEST) REFERENCES Quesito(ID, TITOLO_TEST) ON DELETE CASCADE,
    FOREIGN KEY(ID_TABELLA) REFERENCES Tabella_Esercizio(ID) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Domanda_Chiusa(
    ID_DOMANDA_CHIUSA INT,
    TITOLO_TEST VARCHAR(255),
    PRIMARY KEY (ID_DOMANDA_CHIUSA, TITOLO_TEST),
    FOREIGN KEY(ID_DOMANDA_CHIUSA, TITOLO_TEST) REFERENCES Quesito(ID, TITOLO_TEST) ON DELETE
        CASCADE
)ENGINE=INNODB ;

CREATE TABLE Opzione_Risposta(
    ID INT,
    ID_DOMANDA_CHIUSA INT,
    TITOLO_TEST VARCHAR(255),
    TESTO TEXT NOT NULL,
    SOLUZIONE BOOLEAN NOT NULL,
    PRIMARY KEY(ID, ID_DOMANDA_CHIUSA, TITOLO_TEST),
    FOREIGN KEY(ID_DOMANDA_CHIUSA, TITOLO_TEST) REFERENCES Domanda_Chiusa(ID_DOMANDA_CHIUSA,
        TITOLO_TEST) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Domanda_Codice(
    ID_DOMANDA_CODICE INT,
    TITOLO_TEST VARCHAR(255),
    PRIMARY KEY(ID_DOMANDA_CODICE, TITOLO_TEST),
    FOREIGN KEY(ID_DOMANDA_CODICE, TITOLO_TEST) REFERENCES Quesito(ID, TITOLO_TEST) ON DELETE
        CASCADE
)ENGINE=INNODB ;

CREATE TABLE Sketch_Codice(
    ID INT,
    ID_DOMANDA_CODICE INT,
    TITOLO_TEST VARCHAR(255),
    TESTO TEXT NOT NULL,
    SOLUZIONE BOOLEAN NOT NULL,
```

```
    PRIMARY KEY(ID, ID_DOMANDA_CODICE, TITOLO_TEST),
    FOREIGN KEY(ID_DOMANDA_CODICE, TITOLO_TEST) REFERENCES Domanda_Codice(ID_DOMANDA_CODICE,
        TITOLO_TEST) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Completamento(
    TITOLO_TEST VARCHAR(255),
    EMAIL_STUDENTE VARCHAR(255),
    STATO ENUM('APERTO', 'INCOMPLETAMENTO', 'CONCLUSO') NOT NULL,
    DATA_ULTIMARISPOSTA DATETIME,
    DATA_PRIMARISPOSTA DATETIME,
    PRIMARY KEY(TITOLO_TEST, EMAIL_STUDENTE),
    FOREIGN KEY(TITOLO_TEST) REFERENCES Test(TITOLO) ON DELETE CASCADE,
    FOREIGN KEY(EMAIL_STUDENTE) REFERENCES Studente(EMAIL_STUDENTE) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Risposta(
    EMAIL_STUDENTE VARCHAR(255),
    ID_QUESITO INT,
    TITOLO_TEST VARCHAR(255),
    TESTO TEXT NOT NULL,
    ESITO BOOLEAN NOT NULL,
    PRIMARY KEY(EMAIL_STUDENTE, ID_QUESITO, TITOLO_TEST),
    FOREIGN KEY(EMAIL_STUDENTE) REFERENCES Studente(EMAIL_STUDENTE) ON DELETE CASCADE,
    FOREIGN KEY(ID_QUESITO, TITOLO_TEST) REFERENCES Quesito(ID, TITOLO_TEST) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Messaggio(
    ID INT AUTO_INCREMENT PRIMARY KEY,
    EMAIL_DOCENTE VARCHAR(255) NOT NULL,
    TESTO TEXT NOT NULL,
    TITOLO VARCHAR(255) NOT NULL,
    TITOLO_TEST VARCHAR(255) NOT NULL,
    DATA_INSERIMENTO DATE NOT NULL,
    FOREIGN KEY(TITOLO_TEST) REFERENCES Test(TITOLO) ON DELETE CASCADE,
    FOREIGN KEY(EMAIL_DOCENTE) REFERENCES Docente(EMAIL_DOCENTE) ON DELETE CASCADE
)ENGINE=INNODB ;

CREATE TABLE Messaggio_Studente(
    ID_MESSAGGIO_STUDENTE INT PRIMARY KEY,
    EMAIL_STUDENTE VARCHAR(255) NOT NULL,
    FOREIGN KEY(ID_MESSAGGIO_STUDENTE) REFERENCES Messaggio(ID) ON DELETE CASCADE,
    FOREIGN KEY(EMAIL_STUDENTE) REFERENCES Studente(EMAIL_STUDENTE) ON DELETE CASCADE
)ENGINE=INNODB ;
```

Codice SQL completo delle stored procedure dello schema della basi di dati.

```
USE ESQLDB;

DROP PROCEDURE IF EXISTS Inserimento_Studente;
DROP PROCEDURE IF EXISTS Inserimento_Docente;
DROP PROCEDURE IF EXISTS Inserimento_Tabella_Esercizio;
DROP PROCEDURE IF EXISTS Inserimento_Manipolazione_Riga;
DROP PROCEDURE IF EXISTS Inserimento_Attributo;
DROP PROCEDURE IF EXISTS Inserimento_Quesito;
DROP PROCEDURE IF EXISTS Inserimento_Domanda_Chiusa;
DROP PROCEDURE IF EXISTS Inserimento_Domanda_Codice;
DROP PROCEDURE IF EXISTS Inserimento_Opzione_Risposta;
DROP PROCEDURE IF EXISTS Inserimento_Sketch_Codice;
```

```
DROP PROCEDURE IF EXISTS Inserimento_Risposta;
DROP PROCEDURE IF EXISTS Inserimento_Afferenza;
DROP PROCEDURE IF EXISTS Inserimento_Test;
DROP PROCEDURE IF EXISTS Inserimento_Vincolo_Integrita;
DROP PROCEDURE IF EXISTS Inserimento_Completamento;
DROP PROCEDURE IF EXISTS Inserimento_Messaggio_Docente;
DROP PROCEDURE IF EXISTS Inserimento_Messaggio_Studente;
DROP PROCEDURE IF EXISTS Aggiornamento_Chiave;
DROP PROCEDURE IF EXISTS Aggiornamento_Test;
DROP PROCEDURE IF EXISTS Eliminazione_Tabella_Esercizio;
DROP PROCEDURE IF EXISTS Eliminazione_Manipolazione_Riga;
DROP PROCEDURE IF EXISTS Eliminazione_Quesito;
DROP PROCEDURE IF EXISTS Eliminazione_Test;
DROP PROCEDURE IF EXISTS Eliminazione_Opzione_Risposta;
DROP PROCEDURE IF EXISTS Eliminazione_Sketch_Codice;

DELIMITER |
CREATE PROCEDURE Inserimento_Studente(IN EMAIL VARCHAR(255), IN PSWD VARCHAR(255), IN NOME
    VARCHAR(255), IN COGNOME VARCHAR(255), IN TELEFONO INT(10), IN ANNO_IMMATRICOLAZIONE INT(4),
    IN CODICE VARCHAR(16))
BEGIN
   DECLARE countStudente INT DEFAULT 0;
   SET countStudente=(SELECT COUNT(*) FROM Utente JOIN Studente ON (EMAIL=EMAIL_STUDENTE) WHERE
       (Utente.EMAIL=EMAIL));
   IF (countStudente=0) THEN
      INSERT INTO Utente(EMAIL, PSWD, NOME, COGNOME, TELEFONO) VALUES (EMAIL, PSWD, NOME,
          COGNOME, TELEFONO);
      INSERT INTO Studente(EMAIL_STUDENTE, ANNO_IMMATRICOLAZIONE, CODICE) VALUES (EMAIL,
          ANNO_IMMATRICOLAZIONE, CODICE);
   END IF;
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Docente(IN EMAIL VARCHAR(255), IN PSWD VARCHAR(255), IN NOME
    VARCHAR(255), IN COGNOME VARCHAR(255), IN TELEFONO INT(10), IN NOME_DIPARTIMENTO
    VARCHAR(255), IN NOME_CORSO VARCHAR(16))
BEGIN
   DECLARE countDocente INT DEFAULT 0;
   SET countDocente=(SELECT COUNT(*) FROM Utente JOIN Docente ON (EMAIL=EMAIL_DOCENTE) WHERE
       (Utente.EMAIL=EMAIL));
   IF (countDocente=0) THEN
      INSERT INTO Utente(EMAIL, PSWD, NOME, COGNOME, TELEFONO) VALUES (EMAIL, PSWD, NOME,
          COGNOME, TELEFONO);
      INSERT INTO Docente(EMAIL_DOCENTE, NOME_DIPARTIMENTO, NOME_CORSO) VALUES (EMAIL,
          NOME_DIPARTIMENTO, NOME_CORSO);
   END IF;
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Tabella_Esercizio(IN NOME VARCHAR(255), IN DATA_CREAZIONE DATETIME,
    IN NUM_RIGHE INT, IN EMAIL_DOCENTE VARCHAR(255))
BEGIN
   DECLARE countTabella INT DEFAULT 0;
   SET countTabella=(SELECT COUNT(*) FROM Tabella_Esercizio WHERE (Tabella_Esercizio.NOME=NOME));
   IF (countTabella=0) THEN
```

```
      INSERT INTO Tabella_Esercizio(NOME, DATA_CREAZIONE, NUM_RIGHE, EMAIL_DOCENTE) VALUES (NOME,
          DATA_CREAZIONE, NUM_RIGHE, EMAIL_DOCENTE);
   END IF;
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Manipolazione_Riga(IN ID_TABELLA INT)
BEGIN
   DECLARE countTabella INT DEFAULT 0;
   SET countTabella=(SELECT COUNT(*) FROM Tabella_Esercizio WHERE
       (Tabella_Esercizio.ID=ID_TABELLA));
   IF (countTabella>0) THEN
      INSERT INTO Manipolazione_Riga(ID_TABELLA) VALUES (ID_TABELLA);
   END IF;
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Attributo(IN ID_TABELLA INT, IN TIPO VARCHAR(255), IN NOME
    VARCHAR(255), IN CHIAVE_PRIMARIA BOOLEAN)
BEGIN
   INSERT INTO Attributo(ID, ID_TABELLA, TIPO, NOME, CHIAVE_PRIMARIA) VALUES (NULL, ID_TABELLA,
       TIPO, NOME, CHIAVE_PRIMARIA);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Quesito(IN ID INT, IN TITOLO_TEST VARCHAR(255), IN DIFFICOLTA
    ENUM('BASSO', 'MEDIO', 'ALTO'), IN NUM_RISPOSTE INT, IN DESCRIZIONE VARCHAR(255))
BEGIN
   INSERT INTO Quesito(ID,TITOLO_TEST, DIFFICOLTA, NUM_RISPOSTE, DESCRIZIONE) VALUES
       (ID,TITOLO_TEST, DIFFICOLTA, NUM_RISPOSTE, DESCRIZIONE);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Domanda_Chiusa(IN ID_QUESITO INT, IN TITOLO_TEST VARCHAR(255))
BEGIN
   INSERT INTO Domanda_Chiusa(ID_DOMANDA_CHIUSA, TITOLO_TEST) VALUES (ID_QUESITO, TITOLO_TEST);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Domanda_Codice(IN ID_QUESITO INT, IN TITOLO_TEST VARCHAR(255))
BEGIN
   INSERT INTO Domanda_Codice(ID_DOMANDA_CODICE, TITOLO_TEST) VALUES (ID_QUESITO, TITOLO_TEST);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Opzione_Risposta(IN ID INT, IN ID_DOMANDA_CHIUSA INT, IN
    TITOLO_TEST VARCHAR(255), IN TESTO TEXT, IN SOLUZIONE BOOLEAN)
```

```
BEGIN
   INSERT INTO Opzione_Risposta(ID, ID_DOMANDA_CHIUSA, TITOLO_TEST, TESTO, SOLUZIONE) VALUES
       (ID, ID_DOMANDA_CHIUSA, TITOLO_TEST, TESTO, SOLUZIONE);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Sketch_Codice(IN ID INT, IN ID_DOMANDA_CODICE INT, IN TITOLO_TEST
    VARCHAR(255), IN TESTO TEXT, IN SOLUZIONE BOOLEAN)
BEGIN
   INSERT INTO Sketch_Codice(ID, ID_DOMANDA_CODICE, TITOLO_TEST, TESTO, SOLUZIONE) VALUES (ID,
       ID_DOMANDA_CODICE, TITOLO_TEST, TESTO, SOLUZIONE);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Risposta(IN EMAIL_STUDENTE VARCHAR(255), IN ID_QUESITO INT, IN
    TITOLO_TEST VARCHAR(255), IN TESTO TEXT, IN ESITO BOOLEAN)
BEGIN
   DECLARE countRisposta INT DEFAULT 0;
   SET countRisposta=(SELECT COUNT(*) FROM Risposta WHERE
       (Risposta.EMAIL_STUDENTE=EMAIL_STUDENTE) AND (Risposta.ID_QUESITO=ID_QUESITO) AND
       (Risposta.TITOLO_TEST=TITOLO_TEST));
   IF (countRisposta=0) THEN
      INSERT INTO Risposta(EMAIL_STUDENTE, ID_QUESITO, TITOLO_TEST, TESTO, ESITO) VALUES
          (EMAIL_STUDENTE, ID_QUESITO, TITOLO_TEST, TESTO, ESITO);
   ELSE
      UPDATE Risposta SET Risposta.TESTO=TESTO, Risposta.ESITO=ESITO WHERE
          (Risposta.EMAIL_STUDENTE=EMAIL_STUDENTE) AND (Risposta.ID_QUESITO=ID_QUESITO) AND
          (Risposta.TITOLO_TEST=TITOLO_TEST);
   END IF;
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Afferenza(IN ID_QUESITO INT, IN TITOLO_TEST VARCHAR(255), IN
    ID_TABELLA INT)
BEGIN
   INSERT INTO Afferenza(ID_QUESITO, TITOLO_TEST, ID_TABELLA) VALUES (ID_QUESITO, TITOLO_TEST,
       ID_TABELLA);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Test(IN TITOLO VARCHAR(255), IN EMAIL_DOCENTE VARCHAR(255), IN FOTO
    LONGBLOB, IN DATA_CREAZIONE DATE, IN VISUALIZZA_RISPOSTE BOOLEAN)
BEGIN
   INSERT INTO Test(TITOLO, EMAIL_DOCENTE, FOTO, DATA_CREAZIONE, VISUALIZZA_RISPOSTE) VALUES
       (TITOLO, EMAIL_DOCENTE, FOTO, DATA_CREAZIONE, VISUALIZZA_RISPOSTE);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Completamento(IN TITOLO_TEST VARCHAR(255), IN EMAIL_STUDENTE
```

```
    VARCHAR(255))
BEGIN
   DECLARE countInsert INT DEFAULT 0;
   SET countInsert=(SELECT COUNT(*) FROM Completamento WHERE
       (Completamento.TITOLO_TEST=TITOLO_TEST) AND
       (Completamento.EMAIL_STUDENTE=EMAIL_STUDENTE));
   IF (countInsert=0) THEN
      INSERT INTO Completamento(TITOLO_TEST, EMAIL_STUDENTE, STATO) VALUES (TITOLO_TEST,
          EMAIL_STUDENTE, 'APERTO');
   END IF;
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Vincolo_Integrita(IN ID_ATTRIBUTO_REFERENZIANTE INT, IN
    ID_ATTRIBUTO_REFERENZIATO INT)
BEGIN
   INSERT INTO Vincolo_Integrita(REFERENTE, REFERENZIATO) VALUES(ID_ATTRIBUTO_REFERENZIANTE,
       ID_ATTRIBUTO_REFERENZIATO);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Messaggio_Docente(IN EMAIL_DOCENTE VARCHAR(255), IN TESTO TEXT, IN
    TITOLO VARCHAR(255), IN TITOLO_TEST VARCHAR(255), IN DATA_INSERIMENTO DATE)
BEGIN
   INSERT INTO Messaggio(EMAIL_DOCENTE, TESTO, TITOLO, TITOLO_TEST, DATA_INSERIMENTO)
       VALUES(EMAIL_DOCENTE, TESTO, TITOLO, TITOLO_TEST, DATA_INSERIMENTO);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Inserimento_Messaggio_Studente(IN ID_MESSAGGIO_STUDENTE INT, IN EMAIL_STUDENTE
    VARCHAR(255))
BEGIN
   INSERT INTO Messaggio_Studente(ID_MESSAGGIO_STUDENTE, EMAIL_STUDENTE)
       VALUES(ID_MESSAGGIO_STUDENTE, EMAIL_STUDENTE);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Aggiornamento_Chiave(IN ID_TABELLA INT, IN NOME VARCHAR(255))
BEGIN
   UPDATE Attributo SET CHIAVE_PRIMARIA=1 WHERE (Attributo.ID_TABELLA=ID_TABELLA) AND
       (Attributo.NOME=NOME);
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Aggiornamento_Test(IN TITOLO VARCHAR(255))
BEGIN
   DECLARE countTest INT DEFAULT 0;
    DECLARE oldValue INT DEFAULT 0;
   SET countTest=(SELECT COUNT(*) FROM Test WHERE (Test.TITOLO=TITOLO));
```

```
   IF (countTest>0) THEN
      SET oldValue = (SELECT VISUALIZZA_RISPOSTE FROM Test WHERE (Test.TITOLO=TITOLO));
      UPDATE Test SET VISUALIZZA_RISPOSTE=((oldValue+1)%2) WHERE (Test.TITOLO=TITOLO);
   END IF;
END;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Eliminazione_Tabella_Esercizio(IN ID_TABELLA INT)
BEGIN
   DECLARE countTabella INT DEFAULT 0;
   SET countTabella=(SELECT COUNT(*) FROM Tabella_Esercizio WHERE
       (Tabella_Esercizio.ID=ID_TABELLA));
   IF (countTabella>0) THEN
      DELETE FROM Tabella_Esercizio WHERE (Tabella_Esercizio.ID=ID_TABELLA);
   END IF;
END ;
|
DELIMITER ;


DELIMITER |
CREATE PROCEDURE Eliminazione_Manipolazione_Riga(IN ID_TABELLA INT)
BEGIN
   DECLARE countTabella INT DEFAULT 0;
   SET countTabella=(SELECT COUNT(*) FROM Tabella_Esercizio WHERE
       (Tabella_Esercizio.ID=ID_TABELLA));
   IF (countTabella>0) THEN
      DELETE FROM Manipolazione_Riga WHERE (Manipolazione_Riga.ID_TABELLA=ID_TABELLA) LIMIT 1;
   END IF;
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Eliminazione_Quesito(IN ID_QUESITO INT, IN TITOLO_TEST VARCHAR(255))
BEGIN
   DECLARE countQuestion INT DEFAULT 0;
   SET countQuestion=(SELECT COUNT(*) FROM Quesito WHERE (Quesito.ID=ID_QUESITO) AND
       (Quesito.TITOLO_TEST=TITOLO_TEST));
   IF (countQuestion>0) THEN
      DELETE FROM Quesito WHERE (Quesito.ID=ID_QUESITO) AND (Quesito.TITOLO_TEST=TITOLO_TEST);
   END IF;
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Eliminazione_Test(IN TITOLO VARCHAR(255))
BEGIN
   DECLARE countTest INT DEFAULT 0;
   SET countTest=(SELECT COUNT(*) FROM Test WHERE (Test.TITOLO=TITOLO));
   IF (countTest>0) THEN
      DELETE FROM Test WHERE (Test.TITOLO=TITOLO);
   END IF;
END ;
|
DELIMITER ;
```

```
DELIMITER |
CREATE PROCEDURE Eliminazione_Opzione_Risposta(IN ID INT, IN ID_QUESITO INT, IN TITOLO_TEST
    VARCHAR(255))
BEGIN
   DELETE FROM Opzione_Risposta WHERE ((Opzione_Risposta.ID=ID) AND
       (Opzione_Risposta.ID_DOMANDA_CHIUSA=ID_QUESITO) AND
       (Opzione_Risposta.TITOLO_TEST=TITOLO_TEST));
END ;
|
DELIMITER ;

DELIMITER |
CREATE PROCEDURE Eliminazione_Sketch_Codice(IN ID INT, IN ID_QUESITO INT, IN TITOLO_TEST
    VARCHAR(255))
BEGIN
   DELETE FROM Sketch_Codice WHERE ((Sketch_Codice.ID=ID) AND
       (Sketch_Codice.ID_DOMANDA_CODICE=ID_QUESITO) AND (Sketch_Codice.TITOLO_TEST=TITOLO_TEST));
END ;
|
DELIMITER ;
```

Codice SQL completo dei trigger dello schema della basi di dati.

```
USE ESQLDB;

DROP TRIGGER IF EXISTS Inserimento_Record;
DROP TRIGGER IF EXISTS Inserimento_Opzione_Risposta;
DROP TRIGGER IF EXISTS Inserimento_Sketch_Codice;
DROP TRIGGER IF EXISTS Inserimento_Test_Concluso;
DROP TRIGGER IF EXISTS Aggiornamento_Stato_Test;
DROP TRIGGER IF EXISTS Aggiornamento_Test_Concluso;
DROP TRIGGER IF EXISTS Aggiornamento_Visualizza_Risposte;
DROP TRIGGER IF EXISTS Cancellazione_Record;
DROP TRIGGER IF EXISTS Cancellazione_Opzione_Risposta;
DROP TRIGGER IF EXISTS Cancellazione_Sketch_Codice;

DELIMITER |
CREATE TRIGGER Inserimento_Record
AFTER INSERT ON Manipolazione_Riga
FOR EACH ROW
   UPDATE Tabella_Esercizio SET NUM_RIGHE=NUM_RIGHE+1 WHERE (ID=NEW.ID_TABELLA);
|
DELIMITER ;

DELIMITER |
CREATE TRIGGER Inserimento_Opzione_Risposta
AFTER INSERT ON Opzione_Risposta
FOR EACH ROW
   UPDATE Quesito SET NUM_RISPOSTE=NUM_RISPOSTE+1 WHERE (ID=NEW.ID_DOMANDA_CHIUSA) AND
       (TITOLO_TEST=NEW.TITOLO_TEST);
|
DELIMITER ;

DELIMITER |
CREATE TRIGGER Inserimento_Sketch_Codice
AFTER INSERT ON Sketch_Codice
FOR EACH ROW
   UPDATE Quesito SET NUM_RISPOSTE=NUM_RISPOSTE+1 WHERE (ID=NEW.ID_DOMANDA_CODICE) AND
```

```
        (TITOLO_TEST=NEW.TITOLO_TEST);
|
DELIMITER ;

DELIMITER |
CREATE TRIGGER Aggiornamento_Stato_Test
AFTER INSERT ON Risposta
FOR EACH ROW
BEGIN
    DECLARE countQuesiti INT DEFAULT 0;
    DECLARE countRisposteCorrette INT DEFAULT 0;
    DECLARE statoTest VARCHAR(255) DEFAULT NULL;
    SET countQuesiti=(SELECT COUNT(*) FROM Quesito WHERE (Quesito.TITOLO_TEST=NEW.TITOLO_TEST));
    SET countRisposteCorrette=(SELECT COUNT(*) FROM Risposta WHERE
        (Risposta.TITOLO_TEST=NEW.TITOLO_TEST) AND (Risposta.EMAIL_STUDENTE=NEW.EMAIL_STUDENTE)
        AND (Risposta.ESITO=1));
    IF (countQuesiti=countRisposteCorrette) THEN
        UPDATE Completamento SET Completamento.STATO='CONCLUSO',
            Completamento.DATA_ULTIMARISPOSTA=NOW() WHERE
            (Completamento.TITOLO_TEST=NEW.TITOLO_TEST) AND
            (Completamento.EMAIL_STUDENTE=NEW.EMAIL_STUDENTE);
    ELSE
        SET statoTest=(SELECT STATO FROM Completamento WHERE
            (Completamento.TITOLO_TEST=NEW.TITOLO_TEST) AND (Completamento.EMAIL_STUDENTE =
            NEW.EMAIL_STUDENTE));
        IF (statoTest="APERTO") THEN
            UPDATE Completamento SET Completamento.STATO='INCOMPLETAMENTO',
                Completamento.DATA_PRIMARISPOSTA=NOW() WHERE
                (Completamento.TITOLO_TEST=NEW.TITOLO_TEST) AND
                (Completamento.EMAIL_STUDENTE=NEW.EMAIL_STUDENTE);
        END IF;
    END IF;
END
|
DELIMITER ;

DELIMITER |
CREATE TRIGGER Aggiornamento_Test_Concluso
AFTER UPDATE ON Risposta
FOR EACH ROW
BEGIN
    DECLARE countQuesiti INT DEFAULT 0;
    DECLARE countRisposteCorrette INT DEFAULT 0;
    SET countQuesiti=(SELECT COUNT(*) FROM Quesito WHERE (Quesito.TITOLO_TEST=NEW.TITOLO_TEST));
    SET countRisposteCorrette=(SELECT COUNT(*) FROM Risposta WHERE
        (Risposta.TITOLO_TEST=NEW.TITOLO_TEST) AND (Risposta.EMAIL_STUDENTE=NEW.EMAIL_STUDENTE)
        AND (Risposta.ESITO=1));
    IF (countQuesiti=countRisposteCorrette) THEN
        UPDATE Completamento SET Completamento.STATO='CONCLUSO',
            Completamento.DATA_ULTIMARISPOSTA=NOW() WHERE
            (Completamento.TITOLO_TEST=NEW.TITOLO_TEST) AND
            (Completamento.EMAIL_STUDENTE=NEW.EMAIL_STUDENTE);
    END IF;
END
|
DELIMITER ;

DELIMITER |
CREATE TRIGGER Aggiornamento_Visualizza_Risposte
```

```
AFTER UPDATE ON Test
FOR EACH ROW
BEGIN
    DECLARE viewAnswer BOOLEAN DEFAULT 0;
    SET viewAnswer=(SELECT VISUALIZZA_RISPOSTE FROM Test WHERE (Test.TITOLO=NEW.TITOLO));
    IF (viewAnswer=1) THEN
        UPDATE Completamento SET Completamento.STATO='CONCLUSO' WHERE
            (Completamento.TITOLO_TEST=NEW.TITOLO);
    END IF;
END
|
DELIMITER ;

DELIMITER |
CREATE TRIGGER Cancellazione_Record
AFTER DELETE ON Manipolazione_Riga
FOR EACH ROW
    UPDATE Tabella_Esercizio SET NUM_RIGHE=NUM_RIGHE-1 WHERE (ID=OLD.ID_TABELLA);
|
DELIMITER ;

DELIMITER |
CREATE TRIGGER Cancellazione_Opzione_Risposta
AFTER DELETE ON Opzione_Risposta
FOR EACH ROW
    UPDATE Quesito SET NUM_RISPOSTE=NUM_RISPOSTE-1 WHERE (ID=OLD.ID_DOMANDA_CHIUSA);
|
DELIMITER ;

DELIMITER |
CREATE TRIGGER Cancellazione_Sketch_Codice
AFTER DELETE ON Sketch_Codice
FOR EACH ROW
    UPDATE Quesito SET NUM_RISPOSTE=NUM_RISPOSTE-1 WHERE (ID=OLD.ID_DOMANDA_CODICE);
|
DELIMITER ;
```

Codice SQL completo delle view dello schema della basi di dati.

```
USE ESQLDB;

DROP VIEW IF EXISTS Test_Completati;
DROP VIEW IF EXISTS Risposte_Corrette;
DROP VIEW IF EXISTS Risposte_Inserite;

CREATE VIEW Test_Completati (CODICE, NUMERO) AS
    SELECT CODICE, COUNT(*) AS NUMERO
    FROM Studente, Completamento
    WHERE (Studente.EMAIL_STUDENTE=Completamento.EMAIL_STUDENTE)
      AND (Completamento.STATO='CONCLUSO')
   GROUP BY CODICE
    ORDER BY NUMERO DESC;

CREATE VIEW Risposte_Corrette (CODICE, NUMEROCORR, NUMERORIS, PERC) AS
    SELECT CODICE,
    SUM(CASE WHEN ESITO=TRUE THEN 1 ELSE 0 END) AS NUMEROCORR,
    COUNT(*) AS NUMERORIS,
    (SUM(CASE WHEN ESITO=TRUE THEN 1 ELSE 0 END) / COUNT(*)) AS PERC
    FROM Studente, Risposta
```

```
    WHERE (Studente.EMAIL_STUDENTE = Risposta.EMAIL_STUDENTE)
    GROUP BY CODICE
    ORDER BY PERC DESC;

CREATE VIEW Risposte_Inserite (ID_QUESITO, TITOLO_TEST, NUMERO) AS
    SELECT ID_QUESITO, TITOLO_TEST, COUNT(*) AS NUMERO
    FROM Risposta
    GROUP BY ID_QUESITO, TITOLO_TEST
    ORDER BY NUMERO DESC;
```