UNIVERSITÀ DEGLI STUDI FIRENZE

Scuola di Scienze Matematiche Fisiche e Naturali

RELAZIONE BASI DI DATI E SISTEMI INFORMATIVI

DATABASE BAR SCOLASTICO

Membri del progetto:

alessio.majid@stud.unifi.it - 7073646

lorenzo.bartolini8@stud.unifi.it - 7073016

matteo.pascuzzo@stud.unifi.it - 7072913

Informazioni progetto:

Università degli Studi di Firenze

Anno: 2023

Ambiente di sviluppo: MySQL Workbench

Versione: MySQL 8.0

L’IDEA

Il progetto in questione consiste nell'elaborazione e nello sviluppo di un sistema di gestione avanzato per gli acquisti di prodotti gastronomici effettuati presso i bar interni alle strutture scolastiche. Tale sistema mira a fornire una base di dati completa e specifica, in grado di registrare, monitorare e analizzare le transazioni di acquisto effettuate e il saldo degli utenti.

L'obiettivo primario del sistema è consentire ai bar, situati all'interno delle rispettive strutture scolastiche, di gestire in modo efficiente e accurato il loro storico di acquisti. Ciò sarà possibile attraverso l'utilizzo di un applicativo appositamente progettato, che permetterà ai gestori di analizzare i dati e trarre informazioni utili per le proprie attività.

Il database disporrà di diverse funzionalità chiave, tra cui:

1. **Gestione delle transazioni:** Il sistema consentirà di registrare in maniera dettagliata ogni ordine effettuato, tenendo traccia delle informazioni relative alla persona coinvolta (comprensive dei dati personali), alla data dell'acquisto e al bar di riferimento.
   Viene gestito anche il saldo interno all’applicazione da parte dei singoli utenti, infatti è fornita la possibilità di ricaricare il proprio saldo relativo ad un bar all’interno dell’applicativo tramite apposita procedura di ricarica.

2. **Gestione degli ordini:** Il sistema consentirà di gestire eventuali ordini effettuati da parte degli utenti. Tali ordini saranno comprensivi della quantità e pertanto sarà possibile effettuare una ricerca approfondita su tutte le prenotazioni effettuate da un determinato utente.
   Tali prenotazioni sono da intendersi come ordinazioni effettuate prima della ricreazione. Una Volta che comincia la ricreazione gli ordini verranno evasi e perciò confermati.

3. **Gestione dei dettagli:** Il sistema permetterà ai bar di visualizzare e presentare ai propri utenti una lista completa dei prodotti disponibili, fornendo inoltre tutti i dettagli relativi alle vendite di ciascun prodotto. Tra le informazioni visualizzate vi saranno il nome del prodotto, il prezzo e gli eventuali allergeni correlati.

In sintesi, l'obiettivo principale del progetto è creare una piattaforma centralizzata che consenta ai numerosi bar presenti nelle scuole del territorio di gestire in modo efficiente le proprie attività. Inoltre, il sistema permetterà di analizzare approfonditamente i dati relativi agli ordini, ai consumatori e ai prodotti offerti, fornendo informazioni preziose per l'ottimizzazione delle operazioni commerciali.

PROGETTAZIONE CONCETTUALE

Analisi dei requisiti:

Vogliamo prendere in analisi la gestione di una serie di bar facenti parte di istituti comprensivi diversificati che devono gestire una serie di ordini effettuati da persone interne ed esterne (ospiti) alla scuola.
L'utente verrà identificato tramite il proprio cognome, nome, la mail scolastica e una categoria di appartenenza che ne specifica il ruolo all'interno dello stabilimento (classe dello studente, lavoratore scolastico, esterno).
La scuola, nonché la struttura ospitante del bar relativo, è rappresentata univocamente dal proprio codice meccanografico, il nome e la città in cui si trova. Per la scuola è di interesse l'orario in cui finisce la ricreazione per sapere se gli ordini sono da evadere nel giorno stesso o si riferiscono al successivo.
Ogni bar è identificato in modo univoco tramite un codice che, per semplicità, trattandosi di un libero commerciante con licenza verrà descritto come la relativa Partita Iva, inoltre ogni bar metterà a disposizione il proprio numero telefonico, l'email e la scuola di riferimento.
Il bar offrirà quindi una lista di prodotti che potranno essere comprati tramite una transazione digitale. Un determinato utente potrà pertanto effettuare un ordine nel quale specificare il prodotto, la quantità e la data corrente. Si noti come ad ogni ordine corrisponde uno ed un solo prodotto, questo implica che se l'utente intende ordinare diversi prodotti sarà necessario realizzare più ordini separati, uno per ogni prodotto.
Sarà inoltre necessario effettuare una ricarica prepagata da emettere verso il bar della propria struttura studentesca con la quale sarà possibile effettuare tutti i pagamenti e da cui sarà reso disponibile il saldo contabile e il saldo attuale. Non vi è nessun vincolo sul numero di attività che possono far parte di un plesso studentesco. Ovviamente se una scuola fa parte del database significa che almeno un bar è presente al suo interno.
Ogni prodotto all'interno della base di dati sarà contrassegnato dai rispettivi allergeni affinché l'utente possa essere avvertito di eventuali contaminazioni dei prodotti offerti.

DIZIONARI

Dopo un'attenta analisi delle specifiche del progetto e della fase di progettazione concettuale, siamo in grado di catalogare accuratamente tutte le entità coinvolte nel sistema. È fondamentale definire in modo chiaro e dettagliato i relativi attributi, la chiave primaria e i vincoli di integrità associati per ciascuna entità. Di seguito è presente una tabella che illustra le associazioni del progetto, includendo il nome, le entità coinvolte, le cardinalità e gli attributi correlati.

Dizionario delle entità

Nome	Descrizione	Chiave	Attributo
Utente	Utente utilizzatore del servizio. Rappresenta, studenti, personale esterno o professori	Email(pk)	Nome Cognome
Bar	Bar che prestano servizio nelle scuole	PIva(pk)	Email Telefono
Scuola	Struttura che ospita il bar e gli utenti	CodMeccanografico (pk)	Nome Città FineRicreazione
Prodotto	Prodotto venduto da uno specifico bar degli utenti	Id(pk)	Nome Prezzo Allergene
Ordini	Ordine effettuato presso il bar	Id(pk)	Quantità Data Importo
Transazione	Transazione effettuata da un utente verso un bar	Id(pk)	Data Importo Tipo

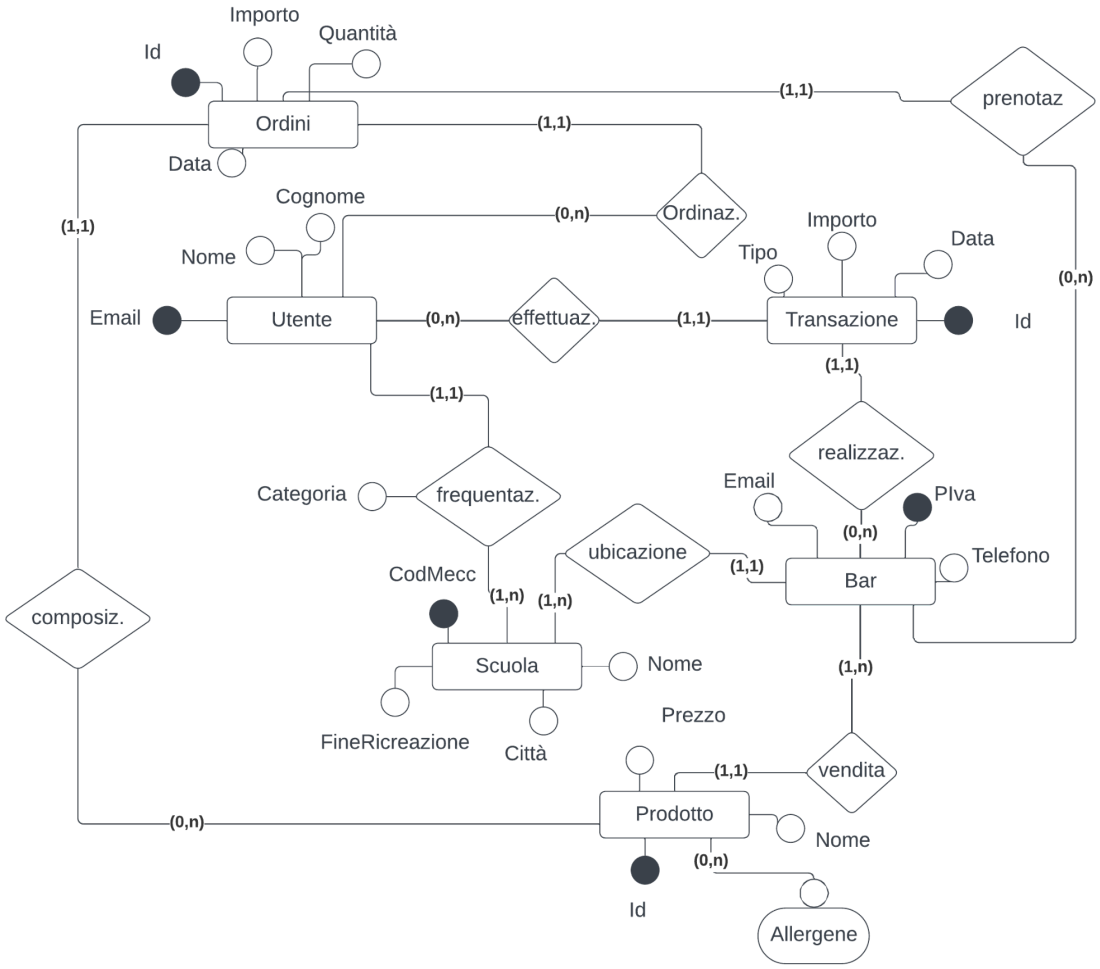
Dizionario delle relazioni

Nome	Descrizione	Entità	Attributo
Prenotazione	Ordini prenotati presso un Bar	Ordini (1,1) Bar (0,n)	X
Ordinazione	Ordinazione di un determinato prodotto da parte di un utente	Utente (0,n) Ordini (1,1)	X
Effettuazione	Transazioni effettuate da un Utente	Utente (0,n) Transazione (1,1)	X
Frequentazione	Frequentazione di un utente ad una determinata Scuola	Utente (1,1) Scuola (1,n)	Categoria
Ubicazione	Ubicazione di un bar all'interno di una scuola	Scuola (1,n) Bar (1,1)	X
Vendita	Lista dei prodotti in vendita da parte dei Bar	Bar (1,n) Prodotto (1,1)	X
Composizione Ordini	Composizione di ordine con il suo prodotto	Ordini (1,1) Prodotto (0,n)	X
Realizzazione	Transazioni realizzate in un bar	Transazione (1,1) Bar (0,n)	X

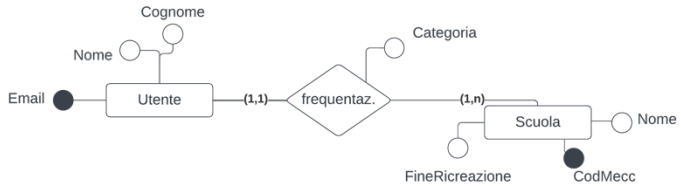
SCHEMA CONCETTUALE

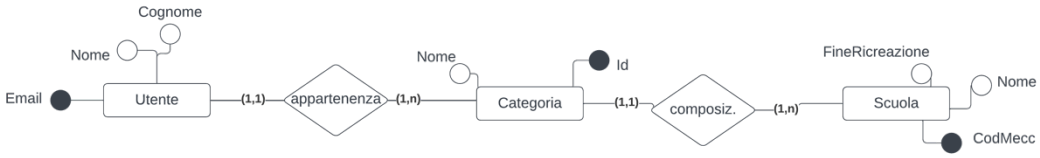
Possiamo procedere alla costruzione di una prima bozza dello schema concettuale, partendo dalla prima entità: il Bar. Tale entità viene caratterizzata da un Id univoco, ovvero la Partita Iva, l'email ed un numero di telefono. L'entità Bar è legata tramite relazioni con l'entità Transazione, Scuola, Ordini e Prodotto. Procedendo con l'entità Transazione, identificata univocamente dal proprio id, possiede gli attributi data, importo e tipo. Sarà legata tramite una relazione con il bar e con l'entità Utente. Quest'ultimo sarà identificato in modo univoco dalla propria mail e sarà descritto ulteriormente tramite gli attributi nome, cognome e categoria di appartenenza della propria scuola. L'entità Scuola è definita in modo univoco tramite il proprio codice Meccanografico e vi sarà la necessità di definire anche la città, il nome e l'orario di fine ricreazione affinché sia possibile verificare l'orario di ritiro dei prodotti acquistati presso un Bar. L'entità Ordini è identificata da un Id progressivo, dalla quantità del prodotto ordinato e dall'importo, infine è di interesse anche la data in cui è stato effettuato l'ordine. Sarà legata tramite una relazione con l'entità Prodotto e con l'Utente che ha eseguito l'ordine. Proseguendo con l'Entità Prodotto, viene caratterizzata univocamente da un Id progressivo, dal suo prezzo e da un attributo che ne descrive la lista degli allergeni.

Di seguito la rappresentazione dello schema E/R:

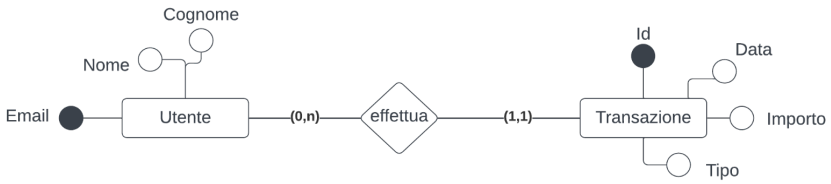


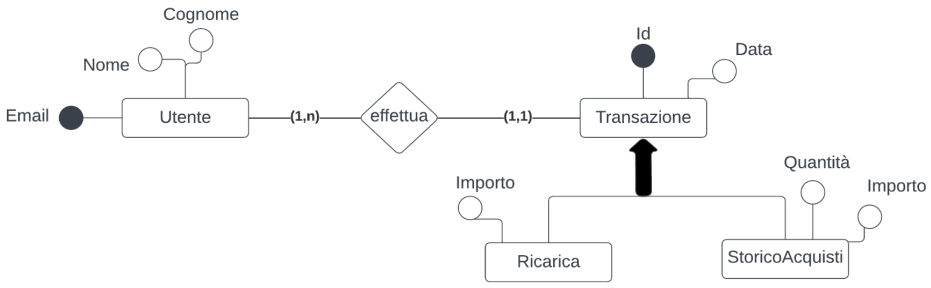
A seguito di questa prima formalizzazione dello schema, notiamo la necessità di reificare la relazione binaria Utente – Frequentazione – Scuola in modo da formalizzare la Categoria di appartenenza dell'utente.

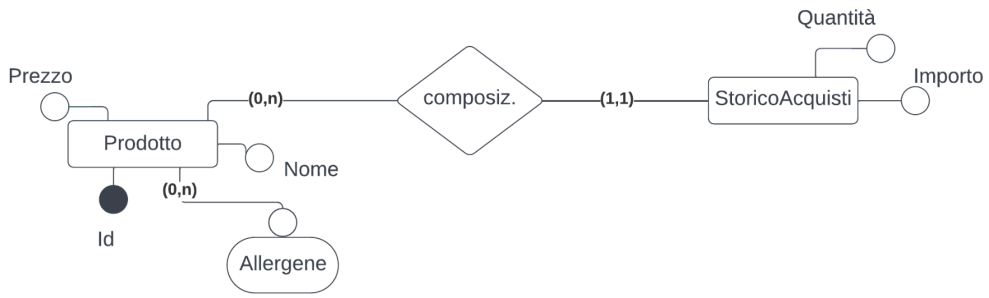




È inoltre nata la necessità di specificare due diversi tipi di Transazioni tramite una generalizzazione: Ricarica e StoricoAcquisti. Facendo tale cambiamento sarà possibile inserire l'importo all'interno delle entità figlie rimuovendolo dall'entità padre. Per quanto riguarda l'entità StoricoAcquisti è necessario memorizzare anche la quantità, con le stesse idee che abbiamo usato per gli ordini, e introdurre una nuova relazione con l'entità Prodotto.







PROGETTAZIONE LOGICA

TABELLA DEI VOLUMI

Si presuppone che il database sia in grado di gestire i bar collocati all'interno di scuole superiori su tutto il territorio fiorentino, il quale consta di circa 60 istituti. Si considera che ogni istituto sia munito di uno o più bar al quale ogni studente può fare richieste di ordinazione e si nota che in media un utente fa 2 ordinazioni al giorno. Il numero medio di studenti che frequentano un istituto è di circa 700, andando a considerare anche insegnanti e lavoratori all'interno dell'istituto possiamo ritenere ragionevole un numero di potenziali utenti per istituto pari a 750 quindi il numero totale di utenti del database sarà circa 45mila. Ogni bar offre intorno ai 30 tipi di prodotto diversi, ognuno con un suo prezzo. Si considera che ogni utente effettui, in media, delle ricariche monetarie 3 volte alla settimana. L'entità Categoria tiene conto delle classi contenute in ogni istituto, dei docenti, dei collaboratori scolastici e degli esterni considerando in media, per ogni scuola, un numero di classi pari a 30, una categoria per i docenti ed una per i collaboratori scolastici.
Per quanto riguarda la relazione Presenza si considera che ogni prodotto possa contenere più di un allergene, considerandone in media 2 si ottiene il numero specificato in tabella.
Le parentesi tonde all'interno della colonna volume specificano che le relazioni sono di tipo (1,1) con l'entità specificata tra parentesi, di conseguenza i loro volumi coincidono.

CONCETTO	TIPO	VOLUME
Bar	E	60
Transazione	E	225000
Acquisti	E	90000
Ricarica	E	135000
Scuola	E	60
Categoria	E	2000
Utente	E	45000
Ordini	E	90000
Prodotto	E	900
Realizzazione	R	225000 (Transazione)
Ubicazione	R	60 (Bar)
Composizione Scuola	R	2000 (Categoria)
Appartenenza	R	45000 (Utente)
Effettuazione	R	225000 (Transazione)

Ordinazione	R	90000 (Ordini)
Prenotazione	R	90000 (Ordini)
Composizione Ordini	R	90000 (Ordini)
Vendita	R	900 (Prodotto)
Composizione Acquisti	R	90000 (Acquisti)

TABELLA DEGLI ACCESSI

Si vuole considerare gli accessi al database fatti in un intervallo di tempo giornaliero. Si presuppone un numero di accessi alla tabella Bar pari alle volte necessarie a reperire un qualunque tipo di informazione riguardo al bar, quali numero di telefono, indirizzo, etc. Nelle Transazioni si considerano, ad esempio, le letture dello storico acquisti oppure tutte le volte in cui si fanno acquisti o ricariche in scrittura.

CONCETTO	COSTRUTTO	ACCESSI	TIPO
Bar	E	10000	Lettura
Transazione	E	100000	Lettura/Scrittura
Acquisti	E	90000	Lettura/Scrittura
Ricarica	E	50000	Lettura/Scrittura
Scuola	E	60	Lettura/Scrittura
Categoria	E	2000	Lettura
Utente	E	300	Lettura
Ordini	E	50000	Lettura/Scrittura
Prodotto	E	60000	Lettura
Realizzazione	R	90000	Lettura/Scrittura
Ubicazione	R	900	Lettura/Scrittura
Composizione Scuola	R	1800	Lettura
Appartenenza	R	280	Lettura
Effettuazione	R	90000	Lettura/Scrittura
Ordinazione	R	80000	Lettura/Scrittura
Prenotazione	R	80000	Lettura/Scrittura
Composizione Prodotto	R	80000	Lettura/Scrittura
Vendita	R	55000	Lettura/Scrittura
Composizione Acquisti	R	80000	Lettura/Scrittura

RISTRUTTURAZIONE SCHEMA ER

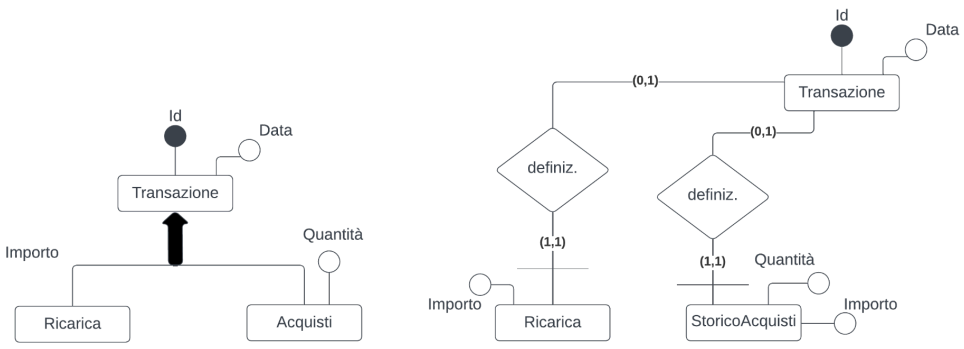
Una volta completato lo schema concettuale, diventa necessario prepararlo per la trasformazione in un progetto logico e successivamente all'implementazione. Ciò richiede la verifica di quattro passaggi che verranno spiegati in dettaglio: analisi delle ridondanze, eliminazione delle generalizzazioni, partizionamento e fusione di concetti, e infine la scelta degli identificatori primari.

Analisi delle ridondanze

L'analisi delle ridondanze implica la rimozione di attributi inutili o dal costo computazionale troppo alto. Osservando lo schema ER, abbiamo notato una ridondanza fra l'entità 'Ordini' e l'entità 'Bar' tramite la relazione Prenotazione. È infatti possibile ricavare il Bar tramite l'entità Prodotto
a scapito di un costo in lettura maggiore. La risoluzione di questa ridondanza risulta necessaria in quanto potrebbero presentarsi delle anomalie, motivo per cui abbiamo deciso di rimuovere l'associazione presente fra le entità 'Ordini' e 'Bar' e lasciare quella fra 'Ordini' e 'Prodotti'.

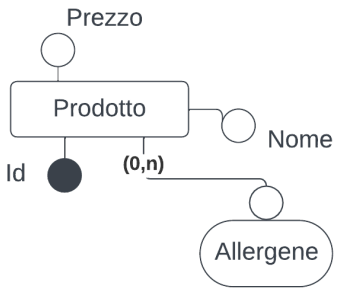
Eliminazione delle generalizzazioni

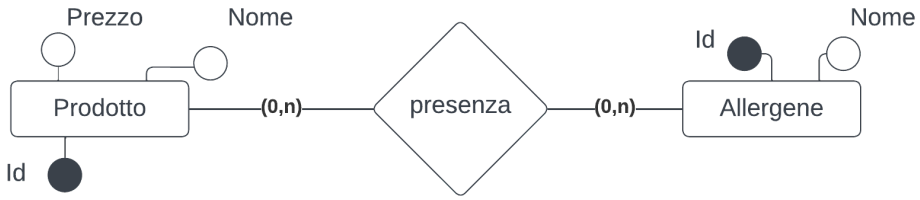
L'eliminazione delle generalizzazioni è stata necessaria a seguito della formalizzazione dell'Entità padre Transazione e delle due entità figlie Ricarica e StoricoAcquisti. Tale generalizzazione è esclusiva totale ed è stato necessario sostituirla con due associazioni e imponendo la chiave di transazione come chiave dei figli.



Partizionamento e accorpamento dei concetti

Nella sezione di partizionamento e accorpamento dei concetti, si verifica che il flusso dei dati sia consono. Durante la progettazione è stata definita l'entità Prodotto descritta da un attributo multi valore chiamato Allergene. Affinché risulti correttamente rappresentato è stato però necessario partizionare tale attributo in un'entità separata, chiamata 'Allergene' legata da una relazione (0,n) con l'entità Prodotto.

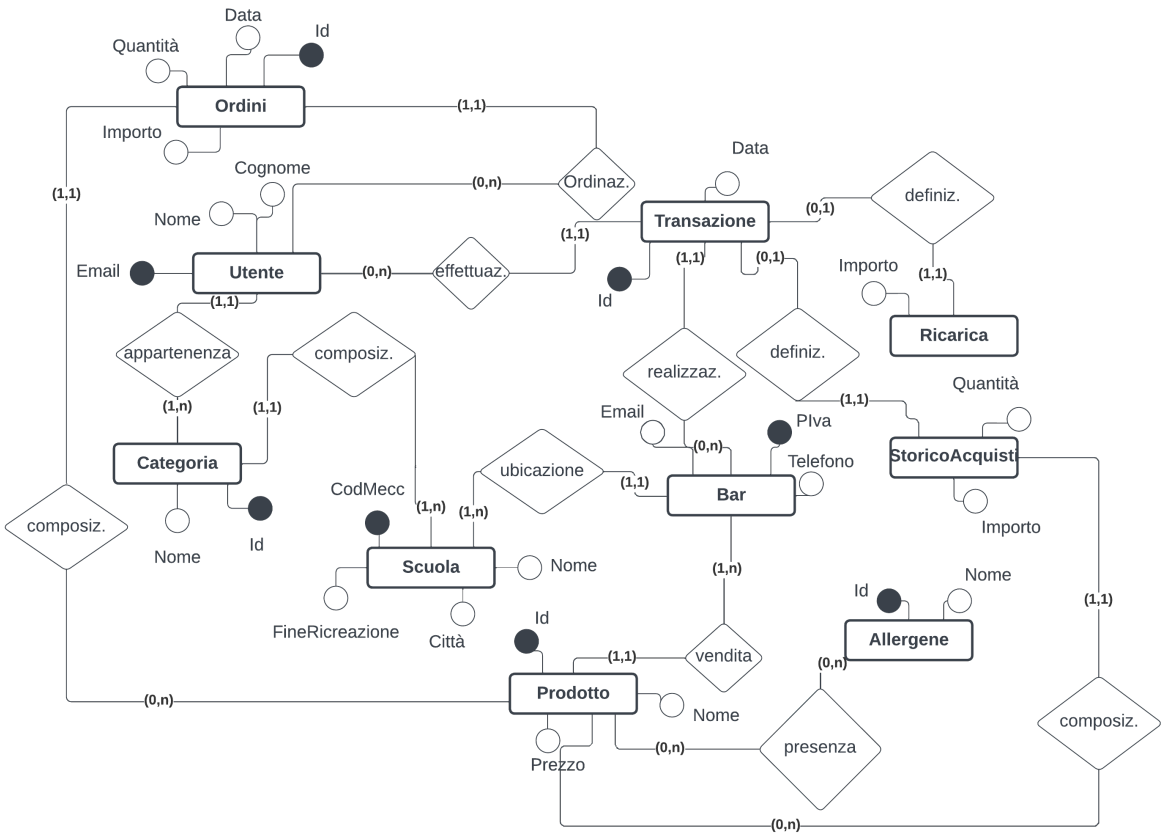




Scelta identificatori primari

Nel contesto specifico del nostro schema concettuale, la scelta degli identificatori si è dimostrata superflua poiché ogni entità aveva già una chiave primaria definita durante la fase di progettazione. Pertanto, non è stato necessario considerare gli identificatori nel processo di transizione al modello relazionale.

Schema finale ristrutturato



TRADUZIONE AL MODELLO LOGICO

Gli attributi sottolineati sono chiavi

Una pratica utile prima di iniziare la fase di traduzione è rappresentare tutte le associazioni e le entità in forma pseudo-tabellare. Questo approccio facilita la visualizzazione chiara di tali informazioni e semplifica il processo di trasformazione. Creando una rappresentazione tabellare, è possibile organizzare in modo strutturato i dettagli delle associazioni e delle entità, rendendo più agevole l'analisi e la comprensione del contesto. Nello specifico il processo di traduzione adottato ci ha permesso di far collassare ogni relazione del tipo (1,1) – (1,n) nell'entità dal lato (1,1). Per esempio la relazione Ordini – Prodotto collassa in modo da definire Prodotto come FK di Ordini.

Si applicano le regole di traduzione al modello logico

Allergene(Id, Nome)

Scuola(CodMeccanografico, Nome, Città, FineRicreazione)

Categoria(Id, Scuola, Nome)
- PK: Id
- Categoria.Scuola → Scuola.CodMeccanografico

Utente(Email, Nome, Cognome, Categoria)
- PK: Email
- Utente.Categoria → Categoria.Id

Bar(PIva, Email, Telefono, Scuola)
- PK: PIva
- Bar.Scuola → Scuola.CodMeccanografico

Prodotto(Id, Bar, Nome, Prezzo, Tipo)
- PK: Id
- Prodotto.Bar → Bar.PIva

PresenzaAllergeneProdotto(Allergene, Prodotto)
- PK: Allergene, Prodotto
- PresenzaAllergeneProdotto.Allergene → Allergene.Id
- PresenzaAllergeneProdotto.Prodotto → Prodotto.Id

Ordini(Id, Utente, Prodotto,, Quantità, Importo)

PK: Id

Ordini.Prodotto → Prodotto.Id

Ordini.Utente → Utente.Email

Transazione(Id, Bar, Utente, Data)

PK: Id

Transazione.Bar → Bar.PIva

Transazione.Utente → Utente.Email

StoricoAcquisti(Transazione, Prodotto, Quantità, Importo)

PK: Transazione

StoricoAcquisti.Transazione → Transazione.Id

StoricoAcquisti.Prodotto → Prodotto.Id

Ricarica(Transazione, Importo)

PK: Transazione

Ricarica.Transazione → Transazione.Id

DIPENDENZE FUNZIONALI

Allergene:

Id → Nome

Scuola:

CodMeccanografico → Nome, Città, FineRicreazione

Categoria:

Id → Scuola, Nome

Utente:

Email → Nome, Cognome, Categoria

Bar:

PIva → Email, Telefono, Scuola

Prodotto:

Id → Bar, Nome, Prezzo, Tipo

PresenzaAllergeneProdotto:

X

Ordini:

Id → Utente, Prodotto, Quantità, Importo

Prodotto, Quantità → Importo

Transazione:

Id → Bar, Utente, Data

StoricoAcquisti:

Transazione → Prodotto, Quantità, Importo

Prodotto, Quantità → Importo

Ricarica

Transazione → Importo

NORMALIZZAZIONE

1FN

Essendo che ogni attributo è definito su un dominio con valori atomici (non si hanno né attributi multi valore né con cardinalità), allora è garantita la 1FN.

2FN

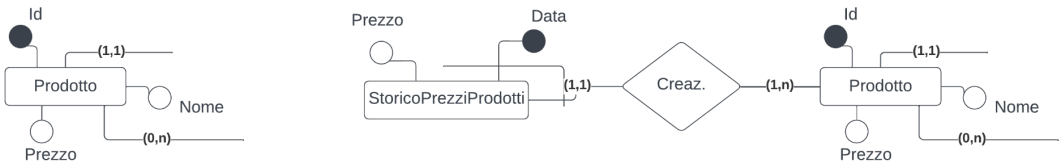
Essendo che tutte le nostre superchiavi sono definite da un attributo singolo non ci sono dipendenze parziali dalla chiave quindi siamo in 2FN.

3FN

E' in terza forma normale se per ogni dipendenza funzionale non banale x → y, x è la superchiave o y è un attributo della chiave.

In questo caso abbiamo rilevato un problema per le tabelle Ordini e StoricoAcquisti.
In entrambi i casi l'importo può essere derivato dal prodotto e dalla quantità calcolandolo in base al prezzo del prodotto.
Abbiamo risolto questa dipendenza rimuovendo l'attributo Importo, sia da Ordini che da StoricoAcquisti.
Questa soluzione è corretta concettualmente ma introduce una nuova problematica. La problematica in questione è la modifica del prezzo del Prodotto. Infatti, in questo modo se un bar decide di modificare il prezzo di un prodotto, la modifica avrà un impatto anche nel passato ovvero negli ordini già effettuati ad un prezzo diverso e per gli acquisti effettuati. Questo è un problema molto importante soprattutto considerato il modo con cui viene calcolato il saldo di un utente, ovvero sommando tutte le ricariche e sottraendo tutti gli acquisti e ordini effettuati.

Per ovviare a ciò abbiamo introdotto la tabella StoricoPrezziProdotti che salverà tutte le modifiche ai prezzi dei singoli prodotti. Questa memorizzazione sarà automatizzata tramite un trigger.
In questo modo sarà sufficiente, usando la funzione GetPrezzoProdotto, ricavare il prezzo del prodotto in base alla data in cui è stato effettuato l'ordine o l'acquisto.
Per facilitare la modifica abbiamo studiato l’inserimento di questa nuova tabella dal punto di vista concettuale modellandolo tramite schema-ER, come mostrato in figura sotto.



Si noti che abbiamo mantenuto l'attributo prezzo in Prodotto. Il significato è quello di prezzo attuale, in questo modo se si intende ricercare il prezzo di un prodotto nel passato si accede a StoricoPrezziProdotti, altrimenti si può interrogare direttamente Prodotto.
Con questa modifica abbiamo perciò aggiunto una nuova tabella StoricoPrezziProdotti e modificato le tabelle Ordini e StoricoAcquisti, definiti come segue:

StoricoPrezziProdotti(Prodotto, Data, Prezzo)
    PK: Prodotto, Data
    StoricoPrezziProdotti.Prodotto → Prodotto.Id
Ordini(Id, Utente, Prodotto,, Quantità)
    PK: Id
    Ordini.Prodotto → Prodotto.Id
    Ordini.Utente → Utente.Email
StoricoAcquisti(Transazione, Prodotto, Quantità)
    PK: Transazione
    StoricoAcquisti.Transazione → Transazione.Id
    StoricoAcquisti.Prodotto → Prodotto.Id

Le nuove dipendenze funzionali sono:

StoricoPrezziProdotti
    Prodotto, Data → Prezzo
Ordini
    Id → Utente, Prodotto, Quantità
StoricoAcquisti
    Transazione → Prodotto, Quantità

Possiamo quindi affermare che abbiamo raggiunto la 3FN.

BCNF

Possiamo affermare che è in BCNF dato che per ogni dipendenza funzionale non banale x → y, x è superchiave. Questo è facilmente osservabile dalla lista delle dipendenze funzionali esposte precedentemente.

IMPLEMENTAZIONE SQL

Lo sviluppo SQL parte dallo schema finale e viene effettuata tramite una serie di fasi progressive che aiutano a garantire la corretta implementazione della base di dati. Le seguenti fasi possono essere riassunte brevemente come segue:

CREAZIONE DEL DATABASE

Al fine di garantire l'assenza di un eventuale database preesistente e obsoleto, adottiamo un'opzione che consente la sua eliminazione per poi ricostruirlo da zero.

INIZIALIZZAZIONE DEL DATABASE

Un'opzione alternativa consiste nell'utilizzare comunque il database, ma eliminando le tabelle al suo interno per poi ricrearle. Questo approccio assicura una corretta visualizzazione del database durante l'esecuzione e consente di evitare modifiche strutturali delle tabelle.
È importante ricordare che le tabelle sono connesse tra loro tramite chiavi esterne (fk), il che richiede un'attenzione specifica nell'ordine di eliminazione delle tabelle in quanto risulta necessario eliminare prima la tabella figlia, che dipende dalla tabella padre, e successivamente eliminare la tabella padre.

CREAZIONE DELLE TABELLE

In questa fase fondamentale si procede alla creazione delle tabelle. Tuttavia, è importante ricordare che l'ordine di creazione delle tabelle è invertito rispetto a quello dell'eliminazione, poiché non sarebbe possibile creare una dipendenza senza avere a disposizione la classe padre che la definisce.

CREAZIONE DEI TRIGGER E DELLE VISTE

Aggiungiamo adesso dei Trigger affinché sia possibile imporre delle condizioni specifiche durante operazioni come l’inserimento o la cancellazione. Creiamo inoltre le viste che ci permetteranno di usufruire dei dati in maniera semplificata.

FASE DI POPOLAMENTO

Durante questa fase, abbiamo inserito nel database dei dati generati casualmente. Anche in questa fase, è importante considerare le dipendenze poiché causerebbe un errore dovuto a una mancanza nei dati. L'inserimento avviene tramite file e tramite codice SQL.

PROCEDURE

Procedura: **EseguiRicarica**

Questa procedura accetta come input un utente, un bar e l'importo. Durante la sua esecuzione inserisce all'interno della tabella 'Transazione' i soggetti presi in causa ed all'interno della tabella 'Ricarica' l'id della transazione e l'importo. Questa procedura permette una facile interazione con il DB.

Procedura: **EseguiAcquisto**

Questa procedura accetta come input un utente, un bar, un prodotto, la quantità e la data. Inserisce all'interno della tabella 'Transazione' i soggetti presi in causa e la data, inoltre inserisce all'interno della tabella 'StoricoAcquisti' l'id della transazione, il prodotto e la quantità acquistata.

Procedura: **ConfermaOrdine**

Questa procedura accetta come input l'id dell'ordine. A seguito del controllo sull'utente viene chiamata la procedure 'EseguiAcquisto' per confermare l'ordine inserendolo nella tabella StoricoAcquisti e, al termine di essa, il record di partenza verrà eliminato dalla tabella 'Ordini'.

FUNZIONI

Funzione: **GetPrezzoProdotto**

Funzione che prende in input un Prodotto e la Data e restituisce un float pari al prezzo basandosi sullo storico contenuto all'interno della tabella "StoricoPrezziProdotto". Viene effettuato un controllo, se la data cercata è più recente dell'ultima data registrata significa che il prezzo cercato è quello attuale del prodotto pertanto viene restituito quello.

Funzione: **BarAccessibileDaUtente**

Funzione che prende in input email dell'utente ed un bar e restituisce il valore booleano True se l'utente è abilitato ad effettuare acquisti presso tale bar, ovvero se il bar è ubicato nella stessa scuola dell'utente, altrimenti False.

Funzione: **GetSaldoUtente**

Funzione che prende in input Email, Bar ed un Flag booleano, per indicare se è richiesto il saldo provvisorio o definitivo, e restituisce il saldo attuale dell'utente calcolandolo come la differenza fra le entrate e le uscite sotto forma rispettivamente di ricariche e acquisti/ordini.

Funzione: **GetSaldoUtenteProvvisorio**

Estensione della funzione "GetSaldoUtente" con cui chiama semplicemente la funzione GetSaldoUtente specificando il Flag correttamente. In questo caso il saldo terrà conto anche degli ordini in sospeso.

Funzione: **GetSaldoUtenteDefinitivo**

Estensione della funzione "GetSaldoUtente" con cui chiama semplicemente la funzione GetSaldoUtente specificando il Flag correttamente. In questo caso il saldo terrà conto solo degli acquisti passati ovvero quelli presenti in StoricoAcquisti.

VISTE

Vista: **OrdiniConImporto**

Vista che restituisce tutti gli ordini effettuati aggiungendo una colonna con l'importo. Utilizza la funzione GetPrezzoProdotto(...) per calcolare l'importo di ogni singolo ordine moltiplicando tale prezzo per la quantità.

Vista: **StoricoAcquistiConImporto**

Vista che restituisce lo storico degli acquisti effettuati aggiungendo una colonna con l'importo. Utilizza la funzione GetPrezzoProdotto(...) per calcolare l'importo di ogni singolo acquisto moltiplicando tale prezzo per la quantità.

Vista: **OrdiniBarAngolo**

Vista che restituisce la lista degli ordini effettuati presso il bar Angolo con pIva pari a 32650198347.

Vista: **TransazioniConImporto**

Vista che restituisce tutte le transazioni effettuate mostrando l'importo. Per farlo esegue una unione fra Ricarica, il cui importo è già noto, e lo StoricoAcquisti tramite la vista StoricoAcquistiConImporto.

TRIGGER

I seguenti trigger sono stati creati al fine di preservare l'integrità dei dati durante il processo di inserimento nelle rispettive tabelle. Essi esaminano le condizioni specificate e, in caso di mancato soddisfacimento, generano segnali di errore per prevenire l'inserimento di dati non validi o non conformi alle regole di integrità.

Trigger: **MemorizzaNuovoPrezzoProdotto**

Questo trigger viene attivato a seguito di una modifica sulla tabella Prodotto. Effettua un controllo sul prezzo e qualora ci sia una differenza fra il vecchio ed il nuovo prezzo, il vecchio valore verrà salvato all'interno della tabella 'StoricoPrezziProdotto' assieme all'id del prodotto e alla data in cui è avvenuta la modifica.

Trigger: **CheckBarInOrdine**

Questo trigger viene attivato prima dell'inserimento di un Ordine presso un bar. Verifica che l'utente possa acquistare presso il suddetto bar tramite la funzione "BarAccessibileDaUtente(...)". Qualora i permessi siano negati, verrà generato un segnale di errore che nega l'inserimento dell'ordine.

Trigger: **CheckBarInTransazione**

Questo trigger viene attivato prima dell'inserimento di una Transazione presso un bar. Verifica che l'utente possa acquistare presso il suddetto bar tramite la funzione "BarAccessibileDaUtente(...)". Qualora i permessi siano negati, verrà generato un segnale di errore che nega l'inserimento dell'ordine