Progetto di Basi di Dati

Cosergate – Gestionale per le spese di casa

Anno Accademico 2012/2013

Daniele Sciuto, Riccardo Serafini, Luca Regazzi, Dario Serra

1. **Raccolta delle specifiche della realtà di interesse espresse in linguaggio naturale, documentata.**

In tutte le case universitarie, durante la settimana i componenti dell'abitazione vanno a fare la spesa.
Non tutti comprano le stesse cose e non tutti consumano le stesse cose.
Nessuno vuole spendere più del necessario, soprattutto gli studenti universitari.
La questione della divisione delle spese, nelle case universitarie, è spesso causa di litigi.
Per risolvere tale inconveniente ci si organizza in modi diversi: c'è chi decide di fare tutta la spesa in comune e cucinare insieme oppure c'è chi decide di organizzarsi autonomamente in modo tale che ognuno compri i propri alimenti e se li cucini.
In entrambi i casi, comunque, compaiono delle voci di spesa che riguardano tutti i componenti dell'abitazione: bollette, prodotti per la pulizia, e in generale tutti i beni di uso comune.
La condivisione di tali beni comporta un periodico rituale di raccolta degli scontrini (spesso dimenticati nei portafogli), trascrizione dei costi e calcolo delle percentuali per definire chi deve soldi a chi.

Secondo noi è molto più comodo affidare la gestione della contabilità casalinga ad un programma. In questo modo si risolvono i problemi descritti e si può beneficiare di ulteriori vantaggi.

Ogni volta che un membro dell'abitazione va a fare la spesa, inserisce nel sistema i prodotti che ha acquistato.
Per ogni prodotto, specifica una breve descrizione, il prezzo, e quali coinquilini lo utilizzeranno.
Il sistema calcola automaticamente quanti soldi ognuno deve restituire agli altri.
Essendo questa procedura così immediata, si può semplicemente evitare di scambiare il denaro ogni volta che si fa la spesa. Visto che di solito si abita con le stesse persone almeno per un anno, si può lasciare il conto "aperto" e restituirsi i soldi solo quando i debiti superano una soglia ragionevole.
Organizzandosi in modo efficace, ruotando le persone che vanno a fare la spesa, si possono tenere basse le differenze tra debiti e crediti.
Se si riesce a tenere questa soglia sotto un certo limite, si può anche evitare del tutto di scambiarsi il denaro.
Oltre a semplificare la procedura descritta in precedenza, l'utilizzo di questo

sistema, può introdurre anche altri vantaggi.
Il programma registra lo storico degli acquisti, questo può essere molto utile per calcolare e presentare delle statistiche sulle spese casalinghe:

- Quanto si spende di media ogni settimana, ogni mese, ogni anno per la casa
- Quali prodotti si comprano più spesso
- Chi va a fare la spesa più spesso
- Monitorare la variazione di prezzo dei prodotti

L'afflusso di una grande quantità di dati relativi ai prodotti venduti nei supermercati, può inoltre rappresentare una risorsa per lo sviluppo di ulteriori funzionalità ed applicazioni.
Associando ad ogni prodotto il suo codice a barre, si può creare con il tempo un database unico di articoli, associando ad ognuno i diversi prezzi proposti da ogni supermercato.
Si potrebbe anche creare una tabella dei supermercati georeferenziata, in modo da mostrare su una mappa quelli più vicini a casa propria, e quelli che offrono prezzi più competitivi.
La raccolta di questi dati potrebbe essere molto facilitata dall'utilizzo di applicazioni per smartphone. Con la fotocamera si registrano i codici a barre dei prodotti nel momento in cui vengono inseriti nel carrello. In questo modo non c'è bisogno di riportare al computer le spese effettuate una volta arrivati a casa.
Il GPS del cellulare, inoltre, può tenere traccia del supermercato in cui ci si trova, e se il prodotto che si sta acquistando è disponibile nelle vicinanze ad un prezzo minore, l'utente può essere avvisato con una notifica.
Il sistema potrebbe aiutare anche a tenere traccia dei prodotti disponibili nella dispensa. Associando la data di scadenza ai generi alimentari, si potrebbe ricevere una notifica quando un prodotto sta per scadere.
Tutte queste funzionalità sono sicuramente ambiziose ma, se il prodotto riesce a raggiungere un discreto numero di utenti, sono certamente realizzabili.
Naturalmente nell'ambito del progetto di Basi di Dati non avremo la possibilità di sviluppare tutte queste funzionalità, ma la nostra fase di Analisi dei Requisiti vuole essere più ampia e completa possibile per facilitare future espansioni.

2. **Analisi dei Requisiti**

Si vuole realizzare una base di dati per gestire le spese all'interno di una abitazione.
La **spesa** è identificata dallo scontrino, in cui è riportata la data e l'ora dell'acquisto, il nome del negozio e a volte anche il nome del cliente, se utilizza una tessera socio.
Ad ogni **prodotto** inserito nel database viene associato un nome, una quantità ed un prezzo, inoltre viene anche associato ai membri dell'abitazione che lo utilizzeranno e quindi concorreranno all'acquisto.
Ogni **abitante** è identificato da un'email, un nome utente, una password, nome e cognome e dalla casa in cui abita.
La **casa** è identificata dalla città e dall'indirizzo, e da un nome identificativo.

Il sistema deve tenere conto della restituzione del denaro tra i componenti dell'abitazione, registrando la data della restituzione, gli abitanti coinvolti e l'importo restituito.
Ad ogni prodotto possono essere aggiunti dei commenti da parte degli utenti, identificati da un contenuto e da una data di inserimento.
Una volta inserite le spese, il sistema dovrà calcolare quanti soldi ogni inquilino deve restituire agli altri.

Specifica delle operazioni

Il sistema deve permettere, e tenere traccia, delle seguenti operazioni:
(il numero di operazioni si riferisce ad un singolo ambiente)

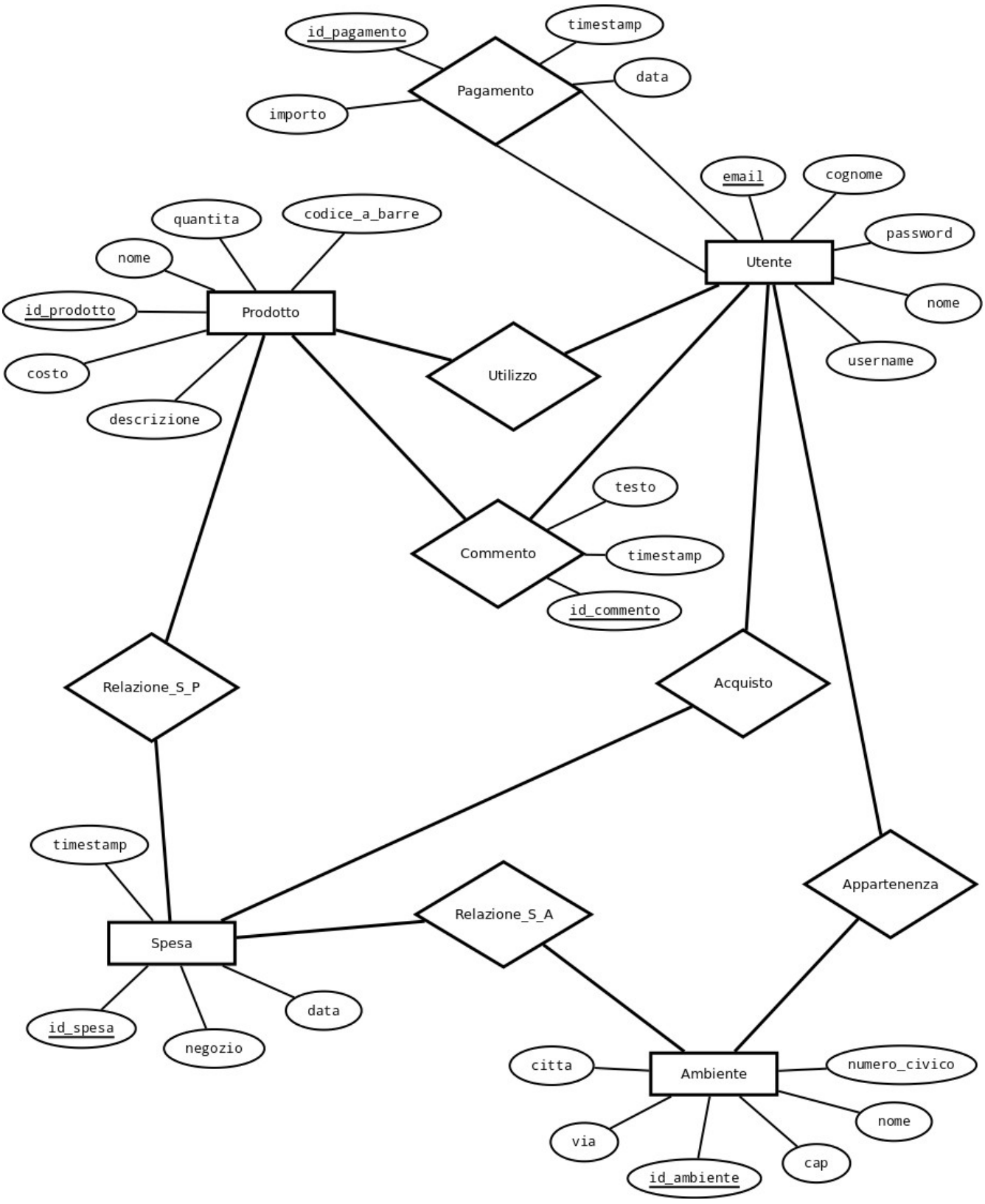
- Creazione/eliminazione di nuovi utenti (raramente)
- Creazione/eliminazione di nuovi ambienti (raramente)
- Inserimento/rimozione di un utente da uno o più ambienti (raramente)
- Inserimento/modifica/rimozione di una nuova spesa (un paio di volte a settimana)
- Inserimento/modifica/rimozione dei prodotti relativi ad una spesa (una ventina volte a settimana)
- Restituzione di denaro tra i componenti della casa (circa una volta al mese)
- Commento ad un prodotto (variabile)

3. **Progettazione concettuale della base di dati con la produzione di uno schema Entity-Relationship (ER), che modelli la realtà di interesse**

Analizzando la realtà di interesse abbiamo isolato diverse entità:
- Prodotto
- Utente
- Spesa
- Ambiente (casa)

Abbiamo inoltre riscontrato la necessità delle seguenti relazioni con attributi:
- Pagamento
- Commento

Da queste considerazioni abbiamo ricavato il seguente schema E/R:


id_pagamento
timestamp
Pagamento
data
importo
email
cognome
quantita
codice_a_barre
password
nome
Utente
id_prodotto
Prodotto
nome
Utilizzo
username
costo
descrizione
testo
Commento
timestamp
id_commento
Relazione_S_P
Acquisto
timestamp
Appartenenza
Relazione_S_A
Spesa
data
id_spesa
negozio
citta
Ambiente
numero_civico
nome
via
cap
id_ambiente

Dizionario dei dati

Entità:

Nome entità	Descrizione	Attributi	Identificatore
Utente	Utilizzatore del sistema, inquilino	nome (stringa) cognome (stringa) password (stringa) username (stringa)	Email(stringa)
Prodotto	Il singolo prodotto acquistato in una spesa (riga di uno scontrino)	nome (stringa) quantita (intero) descrizione (stringa) costo (decimale) codice a barre (intero)	Id_spesa(intero)
Spesa	Una spesa effettuata da un utente per la casa, identificata da uno scontrino	negozio (stringa) data (data) timestamp (data)	id_spesa (intero)
Ambiente	L'abitazione per la quale sono effettuate le spese (può anche non essere una casa)	citta (stringa) via (stringa) numero civico (stringa) cap (intero) nome (stringa)	id_ambiente (intero)

Relazioni:

Nome relazione	Descrizione	Entità coinvolte	Attributi
Pagamento	La restituzione di soldi tra due utenti	(ricorsiva) Utente (1,N)	id_pagamento (intero) timestamp (data) importo (decimale) data (data)
Commento	Un commento effettuato da un utente ad un prodotto	Prodotto (1,N) Utente (1,N)	id_commento (intero) timestamp (data) testo (stringa)
Relazione_S_A	Relazione per associare ogni spesa ad un ambiente	Spesa (1,1) Ambiente (1,N)	

Nome relazione	Descrizione	Entità coinvolte	Attributi
Relazione_S_P	Relazione per associare ogni prodotto alla propria spesa	Prodotto (1,N) Spesa (1,1)	
Appartenenza	Relazione per associare ogni utente ai propri ambiente	Utente (1,N) Ambiente (1,N)	
Acquisto	Relazione per associare ogni spesa all'utente che l'ha effettuata	Utente (1,N) Spesa (1,N)	
Utilizzo	Relazione per associare ogni prodotto all'utente che lo utilizza e partecipa all'acquisto	Prodotto (1,N) Utente (1,N)	

4. **Progettazione logica**

Ristrutturazione dello schema concettuale

Eliminazione delle ridondanze:
non abbiamo riscontrato ridondanze nello schema concettuale, e non riteniamo opportuno introdurne.

Eliminazione delle gerarchie:
non abbiamo riscontrato gerarchie nello schema concettuale.

Accorpamenti:
Si nota che le Relazioni S_A e S_P (spesa ambiente e spesa prodotto) possono essere accorpate a Spesa, in quanto sono relazioni (1,1)(1,N).
Basta semplicemente aggiungere a Spesa le chiavi esterne di Prodotto e Ambiente.
In questo modo tra Spesa e Ambiente, e tra Spesa e Prodotto ci saranno relazioni (1,N).

Ecco come appare lo schema E/R finale:

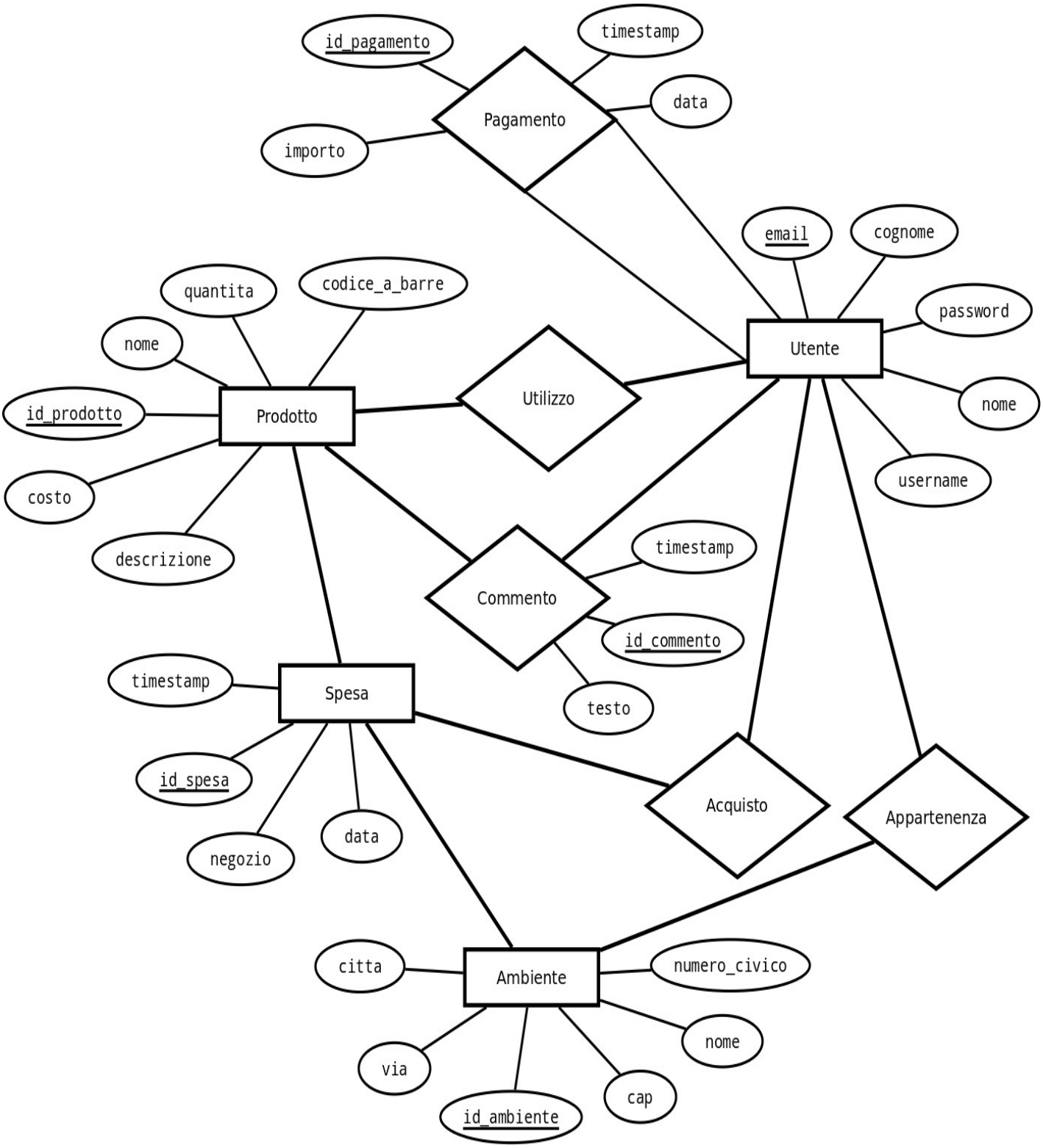

id_pagamento
timestamp
Pagamento
data
importo
email
cognome
password
Utente
nome
username
quantita
codice_a_barre
nome
Utilizzo
id_prodotto
Prodotto
costo
descrizione
timestamp
Commento
id_commento
testo
timestamp
Spesa
id_spesa
negozio
data
Acquisto
Appartenenza
citta
Ambiente
numero_civico
nome
via
cap
id_ambiente

Tavola dei volumi e delle operazioni

Concetto	Tipo	Volume
Prodotto	E	580 x N x anno
Spesa	E	60 X N X anno
Ambiente	E	N
Pagamento	R	20 x N x anno
Commento	R	100 x N x anno
Appartenenza	R	4 x N
Acquisto	R	60 x N x anno
Utilizzo	R	1000 x N x anno

Elenco degli identificatori principali:

Nome entità	Attributo chiave principale
Ambiente	id_ambiente (intero)
Utente	email (stringa)
Prodotto	id_prodotto (intero)
Spesa	id_spesa (intero)

Traduzione dello schema concettuale nello schema relazionale

Entità-Relazione	**Traduzione**	**Vincolo di integrità referenziale**
Ambiente	ambiente(id_ambiente, citta, via, cap, numero_civico, nome)	
Utente	utente(email, nome, cognome, password, username)	
Prodotto	prodotto(id_prodotto, nome, quantita, costo, descrizione, codice_a_barre, spesa)	spesa → spesa.id_spesa
Spesa	spesa(id_spesa, negozio, data, timestamp, ambiente, cliente)	ambiente → ambiente.id_ambiente cliente → utente.email
Appartenenza	appartenenza(id_ambiente, id_utente)	id_ambiente → ambiente.id_ambiente id_utente → utente.email
Pagamento	pagamento(id_pagamento, importo, data, timestamp, id_pagante, id_creditore)	id_pagante → utente.email id_creditore → utente.email
Utilizzo	utilizzo(prodotto, utente)	prodotto → prodotto.id_prodotto utente → utente.email
Commento	commento(id_commento, testo, timestamp, id_prodotto, id_utente)	id_prodotto → prodotto.id_prodotto id_utente → utente.email

Codice SQL:

```
CREATE TABLE IF NOT EXISTS ambiente (
  id_ambiente INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,
  citta VARCHAR(45),
  via VARCHAR(45),
  cap VARCHAR(45),
  numero_civico VARCHAR(45),
  nome VARCHAR(45) NOT NULL
);

CREATE TABLE utente (
  email VARCHAR(45) NOT NULL PRIMARY KEY,
  nome VARCHAR(45) NOT NULL ,
  cognome VARCHAR(45) NOT NULL ,
  password VARCHAR(45) NOT NULL ,
  username VARCHAR(45) NOT NULL
);

CREATE TABLE IF NOT EXISTS appartenenza (
  id_ambiente INTEGER NOT NULL ,
  id_utente VARCHAR(45) NOT NULL ,
  PRIMARY KEY (id_ambiente, id_utente) ,
  FOREIGN KEY (id_ambiente) REFERENCES
ambiente(id_ambiente),
  FOREIGN KEY (id_utente) REFERENCES utente(email)
);

CREATE TABLE IF NOT EXISTS spesa (
  id_spesa INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,
  negozio VARCHAR(45) NULL ,
  data DATE NOT NULL ,
  timestamp DATE NOT NULL ,
  ambiente INTEGER NOT NULL,
  cliente VARCHAR(45) NOT NULL,
  FOREIGN KEY(ambiente) REFERENCES ambiente(id_ambiente),
  FOREIGN KEY(cliente) REFERENCES utente(email)
);
```

```
CREATE TABLE IF NOT EXISTS pagamento (
  id_pagamento INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,
  importo INTEGER NOT NULL ,
  data INTEGER NOT NULL ,
  timestamp INTEGER NOT NULL ,
  id_pagante VARCHAR(45) NOT NULL ,
  id_creditore VARCHAR(45) NOT NULL ,
  FOREIGN KEY (id_pagante) REFERENCES utente(email),
  FOREIGN KEY (id_creditore) REFERENCES utente(email)
);

CREATE TABLE IF NOT EXISTS prodotto (
  id_prodotto INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,
  nome VARCHAR(45) NOT NULL ,
  quantita INTEGER NOT NULL ,
  costo INTEGER NOT NULL ,
  descrizione VARCHAR(45) ,
  codice_a_barre INTEGER ,
  spesa INTEGER NOT NULL ,
  FOREIGN KEY (spesa) REFERENCES spesa(id_spesa)
);

CREATE TABLE IF NOT EXISTS utilizzo (
  prodotto INTEGER NOT NULL ,
  utente VARCHAR(45) NOT NULL ,
  PRIMARY KEY (`prodotto`, `utente`) ,
  FOREIGN KEY (prodotto) REFERENCES prodotto(id_prodotto),
  FOREIGN KEY (utente) REFERENCES utente(email)
);

CREATE TABLE IF NOT EXISTS commento (
  id_commento INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,
  testo INTEGER NOT NULL ,
  timestamp INTEGER NOT NULL ,
  id_prodotto INTEGER NOT NULL ,
  id_utente VARCHAR(45) NOT NULL ,
  FOREIGN KEY (id_prodotto) REFERENCES
prodotto(id_prodotto),
  FOREIGN KEY (id_utente) REFERENCES utente(email)
);
```