Progetto di Basi di Dati

Database per la gestione di un bed and breakfast

Giuseppe Alecci
Matricola 1000001199

Indice

1 Introduzione

Nel seguente elaborato viene descritto il procedimento messo in atto per la creazione di un database per la gestione di un bed and breakfast, partendo dalla fase di progettazione per arrivare poi all'implementazione vera e propria del database.
Vengono utilizzate tecniche di progettazione concettuale, logica e fisica che, partendo da un insieme di requisiti scritti in linguaggio naturale, permettono di creare lo schema logico che verrà implementato.

2 Progettazione concettuale

È la fase iniziale della progettazione che ha lo scopo di raccogliere i requisiti richiesti dal cliente, inizialmente espressi in linguaggio naturale, e di formalizzarli, descrivendoli in maniera sufficientemente astratta da essere indipendenti dal DBMS che sarà usato per l'implementazione del database.

2.1 Requisiti

Si vuole realizzare una base di dati per la gestione di un bed and breakfast che permetta agli ospiti di prenotare una o più camere per un dato lasso di tempo.

Gli ospiti devono fornire al bed and breakfast informazioni quali nome, cognome, indirizzo e numero di telefono. Bisogna tenere conto anche di informazioni sulla prenotazione, come chi ha prenotato, la data di arrivo e di partenza, la stanza prenotata e il prezzo.

Ogni camera deve essere identificata da un numero. L'albergatore deve fornire all'ospite in fase di prenotazione informazioni sulla camera quali il tipo di camera (singola o doppia), il prezzo a notte e lo stato della camera.

Bisogna tenere traccia di ogni prenotazione distinguibili tra loro con un ID prenotazione. Le prenotazioni devono inoltre fornire informazioni quali prenotazioni effettuate, data di prenotazione, data di arrivo e di partenza, numero di ospiti e la stanza prenotata.

Gli ospiti dovranno, ovviamente, pagare tutti i servizi offerti. Il bed and breakfast terrà quindi traccia dei pagamenti effettuati e da effettuare. Un pagamento deve essere descritto da un id e devono contenere informazioni quali il nome del cliente, lo stato del pagamento, id prenotazione, importo e data pagamento.

2.2 Glossario dei termini

TERMINE	DESCRIZIONE	SINONIMI	LEGAME
Ospite	Cliente del bed and breakfast	Cliente	Camera, prenotazione, fattura
Camera	Stanza da letto dove gli ospiti pernottano	Stanza	ospite
Prenotazione	Impegno a occupare e diritto ad aver riservato un posto in una o più camere del bed and breakfast		Ospite, fattura
Pagamento	Documento rilasciato dall'albergatore al cliente nel quale sono specificati i dati necessari a identificare l'avvenuta operazione di pagamento	Tariffa, fattura	Prenotazione, ospite
Servizi	Cosa offre il bed and breakfast agli ospiti		Camera

2.3 Dati di carattere generale

DATI DI CARATTERE GENERALE
Si vuole realizzare una base di dati per la gestione di un bed and breakfast che permetta agli ospiti di prenotare una o più camere per un dato lasso di tempo. Il prezzo totale della prenotazione dipenderà dal numero di ospiti, dalle camere prenotate e dai servizi offerti.

DATI SUGLI OSPITI
Ogni ospite è identificato da un id ed altre informazioni personali quali nome, cognome, numero di documento di identità, numero di telefono, indirizzo email.

DATI SULLE CAMERE
Ogni camera è identificata da un numero che la contraddistingue. La camera può essere singola o doppia.

DATI SULLE PRENOTAZIONI
Le prenotazioni sono contraddistinte da un id prenotazione, id ospite, data di arrivo, data di partenza, numero ospiti e stanza occupata.

DATI SUI PAGAMENTI
Le fatture, rilasciate dal bed and breakfast al cliente, contengono informazioni sulle transazioni effettuate, come l'id della prenotazione, la data della fattura, l'importo da pagare e lo stato del pagamento (pagato, in attesa). Ogni pagamento è contraddistinta da un id pagamento

DATI SUI SERVIZI
I servizi offerti dal bed and breakfast, come colazione, connessione wifi, parcheggio ecc. Include il nome del servizio e il costo ad esso associato

2.4 Specifiche sulle operazioni

Le operazioni richieste all'interno della base di dati sono le seguenti:

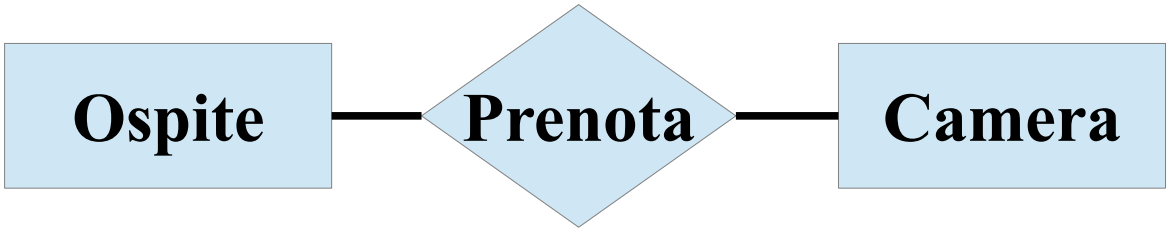
1. Registrazione di un nuovo ospite
2. Modifica dei dati di un ospite
3. Cancellazione di un ospite
4. Registrazione delle prenotazioni
5. Modifica delle prenotazioni
6. Cancellazione delle prenotazioni
7. Registrazione di una camera
8. Controllo disponibilità camere
9. Registrazione del pagamento
10. Registrazione dei servizi
11. Verifica data partenza
12. Verifica numero ospiti

2.5 Strategia di progetto

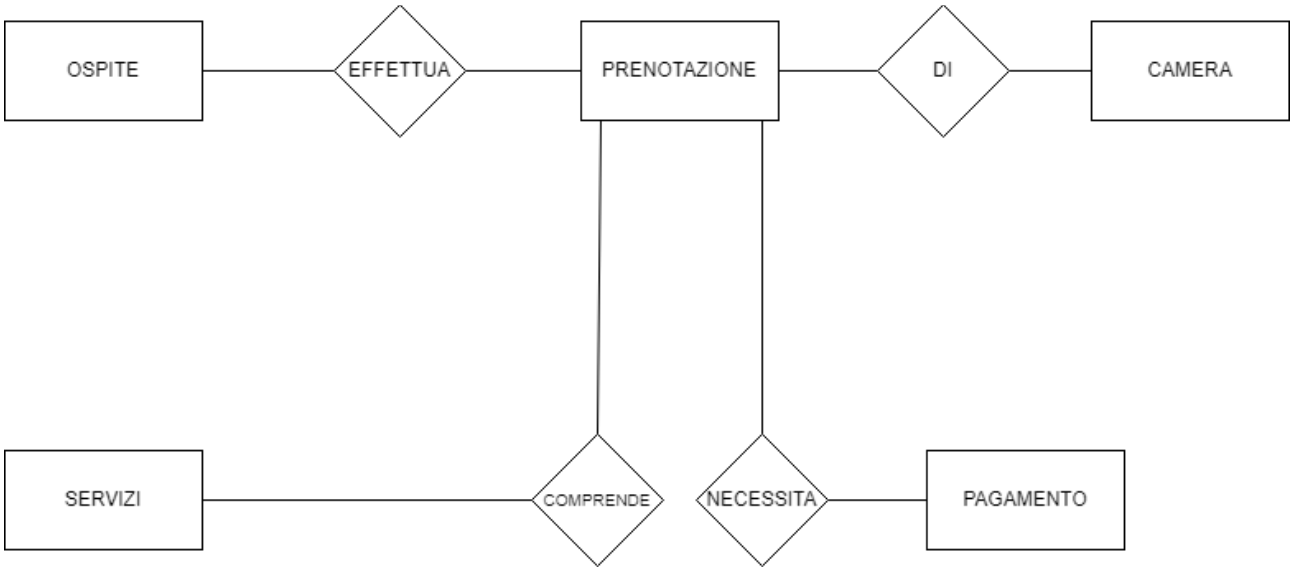
La strategia di progetto indica il modus operandi utilizzato per creare uno schema entità-relazione completo a partire dai requisiti. Le strategie principali sono: top-down, bottom-up e la strategia generale che è una combinazione delle due precedenti.

La strategia che adotteremo, in questo caso, sarà la top-down. La strategia top-down è un approccio che parte dall'alto verso il basso per definire la struttura del database. In pratica, la strategia top-down prevede di definire prima le informazioni generali del sistema e poi di suddividere tali informazioni in componenti più specifici fino a giungere ai dettagli.

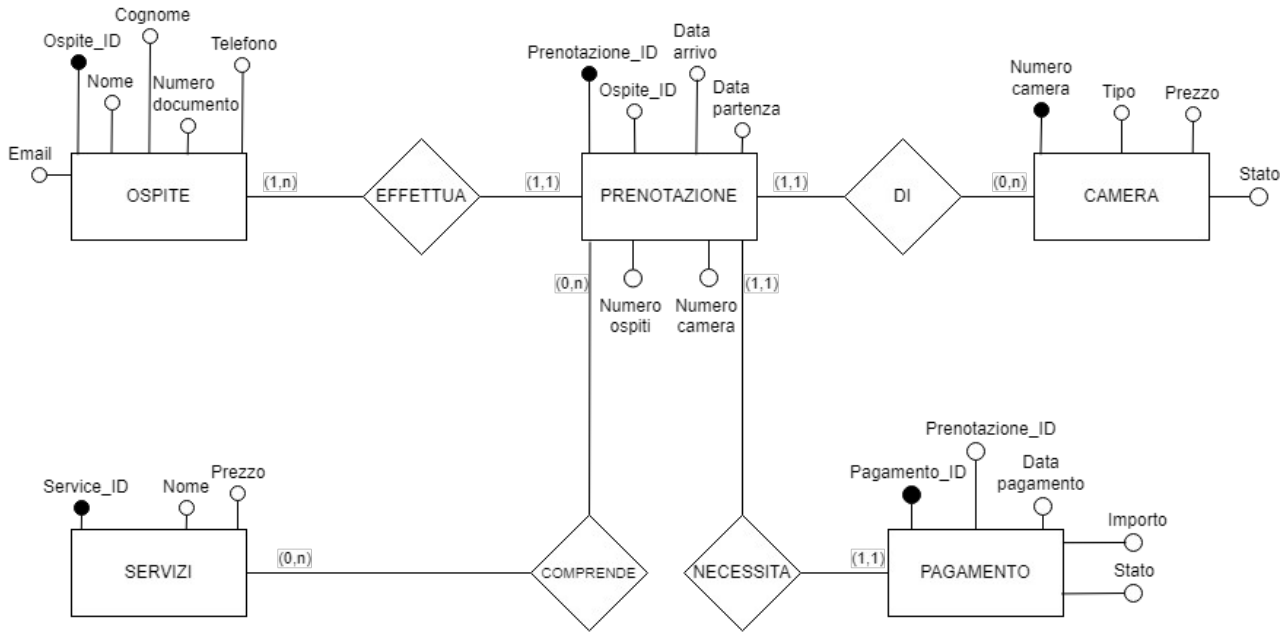
Lo schema scheletro di partenza è il seguente:



Lo schema scheletro viene poi arricchito aggiungendo ulteriori entità e le associazioni che collegano le nuove entità tra loro e con quelle presenti precedentemente nello schema, ottenendo il seguente schema intermedio:



In particolare è stata applicata la trasformazione T4 alla relazione prenotazione rendendola un'entità poiché la relazione descriveva un concetto con esistenza autonoma. Sono stati aggiunti inoltre le entità pagamento, per tenere traccia degli ospiti che hanno pagato il bed and breaskfast, e servizi che tiene traccia dei servizi richiesti dagli ospiti. Viene applicata infine la trasformazione T5 per aggiungere i vari attributi alle entità ottenendo il seguente schema finale:



Lo schema soprastante è quello completo.
Questo però non è in grado di rappresentare tutte le informazioni e i vincoli del sistema richiesto dal cliente, quindi deve essere accompagnato da una documentazione aggiuntiva che contenga tutti i vincoli non esprimibili e le informazioni non presenti nello schema, ma che sono derivabili da quelle rappresentate.
Inoltre bisogna allegare i dizionari di entità e relazioni che permettono di avere informazioni schematizzate e ordinate su tutte le entità e associazioni presenti nello schema.

2.6 Dizionario dei dati – Entità

ENTITÀ	DESCRIZIONE	ATTRIBUTI	IDENTIFICATORI
Ospite	Cliente del bed and breakfast	Ospite_ID, nome, cognome, numero di documento, email, numero di telefono	Ospite_ID
Prenotazione	Impegno a occupare e diritto ad aver riservato un posto in una o più camere del bed and breakfast	Prenotazione_ID, ospite_ID, numero camera, numero ospiti, data arrivo, data partenza	Prenotazione_ID
Camera	Stanza da letto dove gli ospiti pernottano	Numero camera, tipo. Prezzo, stato	Numero camera
Servizi	Cosa offre il bed and breakfast agli ospiti	Service_ID, nome, prezzo	Service_ID
Pagamento	Documento rilasciato dall'albergatore al cliente nel quale sono specificati i dati necessari a identificare l'avvenuta operazione di pagamento	Pagamento_ID, prenotazione_ID, data pagamento, importo, stato	Pagamento_ID

2.7 Dizionario dei dati – relazioni

RELAZIOE	ENTITÀ PARTECIPANTI	DESCRIZIONE	ATTRIBUTI
EFFETTUA	Ospite, prenotazione	Ogni ospite deve effettuare una prenotazione	
DI	Prenotazione, camera	Collega la camera prenotata alla prenotazione corrispondente	
COMPRENDE	Prenotazione, servizi	La prenotazione può comprendere dei servizi	
NECESSITA	Prenotazione, pagamento	Le prenotazioni vanno pagate quando l'ospite lascia il bed and breakfast	

2.8 Vincoli non esprimibili nello schema E-R e dati derivabili

Vincoli non esprimibili

- Una camera non è prenotabile se è occupata;
- Le camere doppie possono ospitare al massimo due persone, le singole una;
- La data di partenza deve essere successiva alla data di arrivo;

Dati derivabili

- La durata del soggiorno può essere ricavata sottraendo la data di partenza alla data di arrivo;
- L'importo totale del soggiorno può essere ricavato moltiplicando la durata dei giorni prenotati per il prezzo giornaliero della camera contando anche i servizi;
- La disponibilità di una camera può essere calcolata confrontando le date di prenotazione esistenti con quelle richieste dal potenziale cliente;

3 Progettazione Logica

È la parte di progettazione che, partendo dallo schema E-R finale ottenuto dalla progettazione concettuale, permette di ottenere lo schema logico che sarà poi implementato.

La progettazione logica si divide in due fasi: la fase di ristrutturazione dello schema E-R e quella di traduzione verso il modello logico.

3.1 Ristrutturazione dello schema E-R

Questa fase è indipendente dal modello logico che si usa per l'implementazione della base di dati e consiste in una sistemazione dello schema E-R. Si calcolano gli indici di prestazione per valutare lo schema (costo di ogni operazione e occupazione della memoria) e sulla base di questi si decide se è necessario fare modifiche e cosa modificare.

La ristrutturazione consta di: analisi delle ridondanze, eliminazione delle generalizzazioni, partizionamento e/o accorpamento di entità e/o associazioni e scelta degli identificatori primari.

La fase preliminare della ristrutturazione dello schema è dunque la creazione delle tabelle dei volumi e delle frequenze, utilizzate poi per effettuare la stima dei costi nelle fasi successive.

3.1.1 Tabella dei volumi

Indica il numero medio di istanze per ogni entità ed associazione che il sistema dovrà gestire.

CONCETTO	TIPO	VOLUME
Ospite	E	60
Prenotazione	E	80
Camera	E	40
Servizi	E	10
Pagamento	E	80
Effettua	R	80
Di	R	80
Comprende	R	240
Necessita	R	80

Il numero delle camere è diviso equamente in 20 singole e 20 doppie.
Per tanto il bed and breakfast potrà avere un massimo di 60 ospiti.
Poiché gli ospiti possono fare più prenotazioni in tempi diversi il numero di prenotazioni è superiore a quello del massimo numero di ospiti che può avere il bed and breaskfast e delle camere disponibili.
Ogni prenotazione include una sola camera. Pertanto, alla relazione *Di* è stato dato lo stesso valore di prenotazioni.
Il bed and breakfast offre diversi servizi. Ogni servizio può essere usato nelle prenotazioni e ogni prenotazioni può avere più servizi. Si è fatta una stima che in media una prenotazione richiede tre servizi e pertanto si è calcolato il valore di 240 per la relazione *Comprende* (80 prenotazioni * 3 servizi).
Poiché ogni prenotazione richiede un pagamento ed ogni pagamento è associato ad una prenotazione il numero di pagamenti e della relazione *Necessita* è uguale al numero delle prenotazioni effettuabili.

3.1.2 Tabella delle frequenze

Indica il tipo di ogni operazione da effettuare nel database (se interattiva o batch) e la sua frequenza.
Le operazioni batch sono eseguite più di rado quindi influiscono meno nella stima dei costi.

OPERAZIONE	TIPO	FREQUENZA
Op. 1	I	20
Op. 2	I	5
Op. 3	I	2
Op. 4	I	40
Op. 5	I	5
Op. 6	I	2
Op. 7	I	40
Op. 8	I	40
Op. 9	I	30
Op. 10	I	20
Op.11	I	40
Op.12	I	40

3.1.3 Analisi delle ridondanze

Nello schema l'unica ridondanza è rappresentata dall'attributo stato nell'entità camere. Esso infatti è un dato che può essere ricavato andando a guardare nelle prenotazioni quali camere sono state prenotate. Analizzeremo ora se è il caso di lasciare questa ridondanza o se sia il caso di eliminarla.

Le operazioni che coinvolgono lo stato di una camera sono la 5, la 7 e la 8.

Per l'operazione 5 in presenza della ridondanza avremo una operazione di scrittura in camere per aggiornare lo stato. Visto che il volume delle camere è 40 e la frequenza dell'operazione 5 è di 5 volte avremo 40*5 = 200 accessi. Se consideriamo che la scrittura ha un costo doppio rispetto alla lettura avremo 200*2 = 400 accessi.

In assenza della ridondanza non dovremmo fare niente per l'operazione 5 quindi avremo 0 accessi.

Per l'operazione 7 sia in presenza della ridondanza sia senza dovremo effettuare comunque una scrittura in camere. Poiché le operazioni da effettuare sono le medesime non c'è bisogno di calcolare il numero di accessi.

Per l'operazione 8 in assenza della ridondanza dovremmo effettuare una lettura delle camere e una lettura delle prenotazioni. Le prenotazioni hanno un volume di 80. per cui avremo 80*40+40*40 = 4800 accessi.

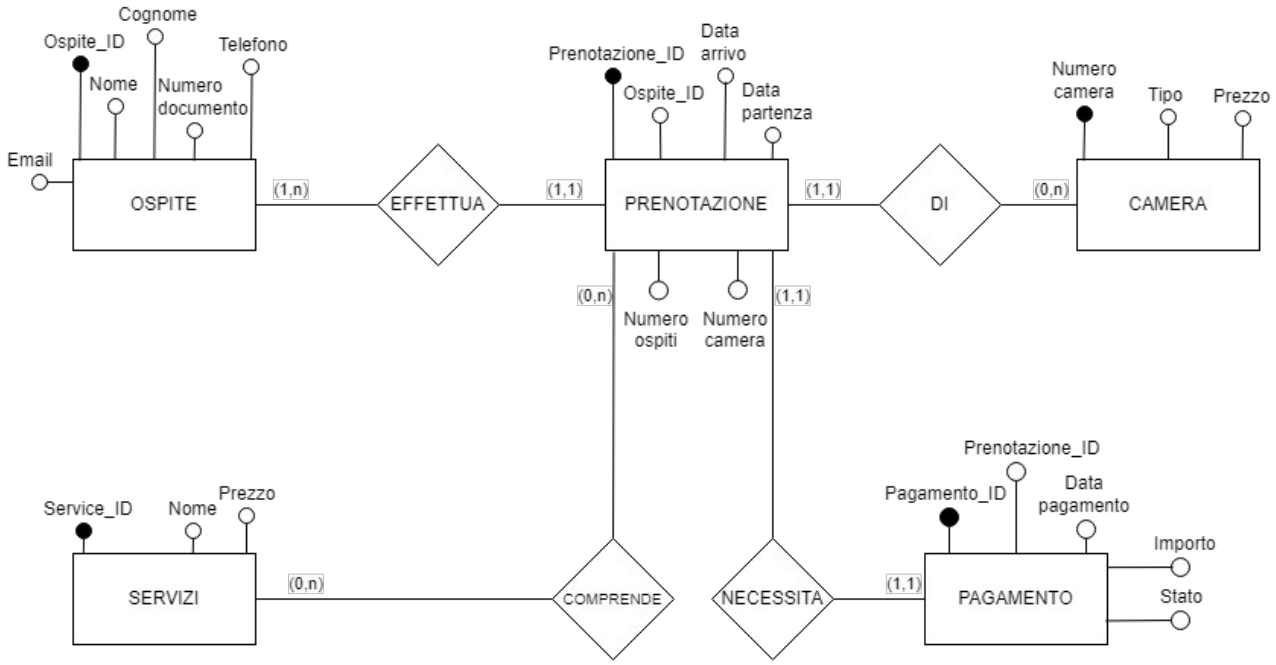
In presenza della ridondanza, invece, basta fare una lettura a camere nel caso si volesse sapere la disponibilità in quel momento. Quindi avremo: volume camere = 40, frequenza operazione 8 = 40. Avremo quindi 1600 accessi. Tuttavia, se volessimo prendere in considerazione la disponibilità in una data precisa dovremo comunque consultare le prenotazioni e quindi avremo anche qui 4800 accessi.

In definitiva se sommiamo tutti gli accessi necessari con la ridondanza, escludendo le operazioni di cui non abbiamo calcolato gli accessi, avremo un totale di 5200 accessi (400 nell'operazione 5 e 4800 nell'operazione 8).

Se sommiamo tutti gli accessi necessari senza ridondanza, escludendo le operazioni di cui non abbiamo calcolato gli accessi, avremo un totale di 4800 accessi (tutti nell'operazione 8).

La conclusione è che è meglio eliminare lo stato.

Il nuovo schema sarà quindi:



3.1.4 Eliminazione delle generalizzazioni

Nelle basi di dati non esistono generalizzazioni, per cui devono essere eliminate dallo schema. I metodi usati per effettuare questo passaggio sono principalmente 3:

- Mantenimento delle entità con associazioni: vengono mantenute tutte le entità, inoltre quelle figlie sono in associazione con quella padre e sono identificate esternamente attraverso l'associazione;
- Collasso verso l'alto: vengono eliminate le entità figlie e le associazioni che le riguardavano si collegano all'entità padre;
- Collasso verso il basso: si elimina l'entità padre e i suoi attributi vengono passati alle entità figlie

Nel nostro caso non ci sono generalizzazioni nello schema E-R finale, dunque questa fase può essere saltata.

3.1.5 Partizionamento/Accorpamento di entità e associazioni

Questo passaggio ha come obiettivo quello di ridurre il più possibile il numero di accessi separando attributi di uno stesso concetto usati da operazioni diverse, o, al contrario, raggruppando attributi di concetti diversi usati dalle stesse operazioni.
Il partizionamento può essere verticale (si creano più entità e gli attributi vengono divisi) o orizzontale (si creano più entità con gli stessi attributi).
La scelta di quale scenario sia più conveniente si fa attraverso la stima dei costi: si calcola il costo delle operazioni prima e dopo il partizionamento (o accorpamento) e si vede quale condizione è la più conveniente.
Nel nostro caso non sono necessari accorpamenti o partizioni.

3.1.6 Scelta degli identificatori primari

Ogni entità necessita di una chiave primaria in modo che il DBMS sappia quali sono le informazioni che identificano univocamente ogni record delle tabelle che si creeranno, in modo da avere un sistema per garantire l'unicità dei valori.
Sono da preferire chiavi semplici (formate da un solo attributo), interne (che non facciano riferimento ad attributi di altre entità) e che contengano attributi che vengono usati spesso per accedere alle istanze di quella particolare entità.
Le chiavi scelte per le entità dello schema E-R sono le seguenti:

- Ospite: ospite_id;
- Prenotazione: prenotazione_id;
- Camera: numero camera;
- Servizi: service_id;
- Pagamento: pagamento_id.

3.2 Traduzione verso il modello logico

È la seconda fase della progettazione logica e consiste nel tradurre le entità e associazioni dello schema E-R nel modello logico scelto (nel nostro caso nel modello relazionale).

A partire dai costrutti dello schema entità-relazione, si creeranno le relazioni del modello relazionale. La distribuzione di chiavi e attributi delle relazioni così create dipende dalle cardinalità che avevano le entità coinvolte nelle associazioni.

Le regole di traduzione principali sono:

- Per associazioni binarie molti a molti, ogni entità e ogni associazione sono tradotte in una relazione avente come attributi gli stessi che aveva il costrutto di partenza. Inoltre le associazioni erediteranno le chiavi primarie delle entità coinvolte;
- Nel caso di associazioni binarie 1 a N, l'associazione viene inglobata nella tabella che rappresenta l'entità che aveva cardinalità (1,1) o (0,1). Questa tabella eredita gli attributi dell'associazione e la chiave dell'altra entità coinvolta;
- Nel caso di associazioni 1 a 1, la traduzione può essere fatta con una, due o tre tabelle a seconda delle cardinalità minime. L'uso di una sola tabella è sconsigliato perché significherebbe annullare scelte di progettazione fatte nelle fasi precedenti.

Il modello relazionale risultante dallo schema E-R finale è il seguente:

- Ospite (ospite_id, nome, cognome, numero documento, telefono, email)
- Prenotazione (prenotazione_id, ospite_id, data arrivo, data partenza, numero ospiti, numero camera)
- Camera (numero camera, tipo, prezzo)
- Servizi (service_id, nome, prezzo)
- Pagamento (pagamento_id, prenotazione_id, data pagamento, importo, stato)
- Comprende(prenotazione_id, service_id)

N.B.: La doppia sottolineatura in alcuni attributi indica che rappresentano sia una chiave interna sia esterna, anche se la rappresentazione corretta sarebbe quella con la sottolineatura tratteggiata per le chiavi esterne.

4 Progettazione fisica

È la fase finale della progettazione, durante la quale il database viene implementato.

Di seguito vengono riportati i codici per la creazione delle tabelle e l'implementazione delle operazioni.

Inoltre verrà allegato un file contenente un database di prova.

4.1 Creazione delle tabelle

- **Tabella Ospite**

```
CREATE TABLE `ospite` (
`ospite_id` int(11) NOT NULL AUTO_INCREMENT,
`nome` varchar(25) NOT NULL,
`cognome` varchar(25) NOT NULL,
`numero_documento` varchar(20) NOT NULL,
`telefono` varchar(10) NOT NULL,
`email` varchar(40) DEFAULT NULL,
PRIMARY KEY (`ospite_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

- **Tabella Prenotazione**

```
CREATE TABLE `prenotazione` (
`prenotazione_id` int(11) NOT NULL AUTO_INCREMENT,
`ospite_id` int(11) NOT NULL,
`data_arrivo` date NOT NULL,
`data_partenza` date NOT NULL,
`numero_ospiti` int(11) NOT NULL,
`numero_camera` int(11) NOT NULL,
PRIMARY KEY (`prenotazione_id`),
CONSTRAINT `prenotazione_ibfk_1` FOREIGN KEY (`ospite_id`)
REFERENCES `ospite` (`ospite_id`) ON DELETE CASCADE ON UPDATE
CASCADE,
CONSTRAINT `prenotazione_ibfk_2` FOREIGN KEY (`numero_camera`)
REFERENCES `camera` (`numero_camera`) ON DELETE CASCADE ON
UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

- **Tabella Camera**

```
CREATE TABLE `camera` (
`numero_camera` int(11) NOT NULL,
`tipo` varchar(10) NOT NULL,
`prezzo` decimal(10,2) NOT NULL,
PRIMARY KEY (`numero_camera`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

- **Tabella Servizi**

```
CREATE TABLE `servizi` (
`service_id` int(11) NOT NULL AUTO_INCREMENT,
`nome` varchar(50) NOT NULL,
`prezzo` decimal(10,2) NOT NULL,
PRIMARY KEY (`service_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

- **Tabella Pagamento**

```
CREATE TABLE `pagamento` (
`pagamento_id` int(11) NOT NULL AUTO_INCREMENT,
`prenotazione_id` int(11) NOT NULL,
`data_pagamento` date DEFAULT NULL,
`importo` decimal(10,2) NOT NULL,
`stato` varchar(20) NOT NULL DEFAULT 'non pagato',
PRIMARY KEY (`pagamento_id`),
CONSTRAINT `pagamento_ibfk_1` FOREIGN KEY (`prenotazione_id`)
REFERENCES `prenotazione` (`prenotazione_id`) ON DELETE CASCADE
ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

- **Tabella Comprende**

```
CREATE TABLE `comprende` (
`service_id` int(11) NOT NULL,
`prenotazione_id` int(11) NOT NULL,
PRIMARY KEY (`service_id`,`prenotazione_id`),
CONSTRAINT `comprende_ibfk_1` FOREIGN KEY (`service_id`)
REFERENCES `servizi` (`service_id`) ON DELETE CASCADE ON
UPDATE CASCADE,
CONSTRAINT `comprende_ibfk_2` FOREIGN KEY (`prenotazione_id`)
REFERENCES `prenotazione` (`prenotazione_id`) ON DELETE CASCADE
ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

4.2 Implementazione delle operazioni

- **Registrazione di un nuovo ospite**

  INSERT INTO ospite (`ospite_id`, `nome`, `cognome`, `numero_documento`, `telefono`, `email`)

  VALUES ($ospite_id, '$nome', '$cognome', '$numero_documento', '$telefono', '$email');

- **Modifica dei dati di un ospite**

  UPDATE `ospite`

  SET `ospite_id`='ospite_id, `nome` = '$nome', `cognome` = '$cognome', `numero_documento` = '$numero_documento', `telefono` = '$telefono', `email` = '$email'

  WHERE `ospite_id` = $ospite_id;

- **Cancellazione di un ospite**

  DELETE FROM `ospite`

  WHERE `ospite_id` = $ospite_id;

- **Registrazione delle prenotazioni**

  INSERT INTO prenotazione (`prenotazione_id`, `ospite_id`, `data_arrivo`, `data_partenza`, `numero_ospiti`, `numero_camera`)

  VALUES ($prenotazione_id, $ospite_id, '$data_arrivo', '$data_partenza', $numero_ospiti, $numero_camera);

  Modifica prenotazioni

```
UPDATE prenotazione
SET `prenotazione_id` = prenotazione_id, `ospite_id` = $ospite_id,
`data_arrivo` = '$data_arrivo', `data_partenza` = '$data_partenza',
`numero_ospiti` = $numero_ospiti, `numero_camera` = $numero_camera
WHERE `prenotazione_id` = $prenotazione_id;
```

- **Cancellazione prenotazione**

```
DELETE FROM `prenotazione`
WHERE `prenotazione_id` = $prenotazione_id;
```

- **Registrazione camera**

```
INSERT INTO camera (`numero_camera`, `tipo`, `prezzo`)
VALUES ($numero_camera, '$tipo', $prezzo);
```

- **Controllo disponibilità camere**

```
CREATE TRIGGER `verifica_numero_ospiti`
BEFORE INSERT ON `prenotazione`
FOR EACH ROW
BEGIN
    DECLARE tipo_camera VARCHAR(10);
    SELECT tipo INTO tipo_camera
    FROM Camera
    WHERE numero_camera = NEW.numero_camera;
    IF (tipo_camera = "singola" AND NEW.numero_ospiti <> 1)
```

```
        OR
        (tipo_camera = "doppia" AND NEW.numero_ospiti > 2) THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = "Numero di
        ospiti non valido";
        END IF;
    END
```

- **Registrazione del pagamento**

```
        INSERT INTO `pagamento` (`pagamento_id`, `prenotazione_id`,
        `data_pagamento`, `importo`, `stato`)
        VALUES ('$pagamento_id','$prenotazione_id', '$data_pagamento', (
        SELECT (DATEDIFF(p.data_partenza, p.data_arrivo) + 1) * (c.prezzo +
        IFNULL(SUM(s.prezzo), 0) )
        FROM prenotazione p JOIN camera c ON p.numero_camera =
        c.numero_camera LEFT JOIN Comprende co  ON p.prenotazione_id =
        co.prenotazione_id LEFT JOIN servizi s ON co.service_id =
        s.service_id
        WHERE p.prenotazione_id = $prenotazione_id
        GROUP BY p.prenotazione_id), '$stato');
```

- **Registrazione servizi**

```
    INSERT INTO servizi (`service_id`, `nome`, `prezzo`)
    VALUES ($service_id, '$nome', $prezzo);
```

- **Verifica data partenza**

```
CREATE TRIGGER verifica_data_partenza
BEFORE INSERT ON prenotazione
FOR EACH ROW
BEGIN
IF NEW.data_partenza < NEW.data_arrivo + INTERVAL 1 DAY THEN
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'La data di partenza deve essere almeno un giorno dopo la data di arrivo.';
END IF;
END
```

- **Verifica numero ospiti**

```
CREATE TRIGGER verifica_numero_ospiti
BEFORE INSERT ON prenotazione
FOR EACH ROW
BEGIN
DECLARE tipo_camera VARCHAR(20);
SELECT tipo INTO tipo_camera
FROM camera
WHERE numero_camera = NEW.numero_camera;
IF (tipo_camera = "singola" AND NEW.numero_ospiti <> 1)
OR
(tipo_camera = "doppia" AND NEW.numero_ospiti > 2)
THEN
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = "Numero di ospiti non valido";
END IF;
END
```