A.D. 1308
unipg
DIPARTIMENTO
DI INGEGNERIA

Base di Dati
Corso di Laurea Ingegneria Informatica ed Elettronica -
A.A. 2024/25

PROGETTO DI UNA BASE DATI

Agenzia di viaggi

Davalli Nicola, Mincigrucci Gabriele, Vescovi Giulia

Sommario

PROGETTAZIONE CONCETTUALE

Raccolta dei requisiti

L'obiettivo del progetto è la creazione di una base di dati per un'agenzia di viaggi che si occupa di organizzare e vendere pacchetti turistici personalizzati, con sede nel comune di Poggiodomo (PG).

I pacchetti offerti dall'agenzia includono una vasta gamma di servizi, come voli, soggiorni in hotel, escursioni, e altre attività. Per garantire un'offerta di qualità, l'agenzia collabora con fornitori esterni, quali compagnie aeree, catene alberghiere e tour operator. La base di dati dovrà essere progettata in modo da raccogliere e organizzare tutte le informazioni necessarie per la gestione dei clienti, delle prenotazioni, dei viaggi e dei fornitori, assicurando una struttura completa e funzionale.

Ogni cliente dell'agenzia dovrà essere identificato da un codice univoco e la base di dati dovrà memorizzare informazioni personali, come il nome, il cognome, la data di nascita, i contatti e l'indirizzo. Sarà inoltre importante tenere traccia delle preferenze di viaggio di ciascun cliente, come il budget disponibile, il clima e il continente preferito. Ogni cliente deve avere la possibilità di lasciare una recensione su un pacchetto indicando quante stelle vuole attribuirgli e la data.

Un ulteriore elemento fondamentale sarà la possibilità di conservare uno storico dettagliato di tutte le prenotazioni effettuate dal cliente. Si prevede di gestire circa 5.000 clienti.

Anche i pacchetti di viaggio offerti dall'agenzia richiedono un'organizzazione all'interno della base di dati. Ogni pacchetto sarà identificato da un codice univoco e sarà necessario registrare informazioni come il nome del pacchetto, la descrizione, la durata, le date di partenza e ritorno, la destinazione principale e il prezzo. Poiché ogni viaggio può includere diversi servizi (ad esempio voli, hotel o escursioni), la base di dati dovrà essere in grado di gestire anche i dettagli di ciascun servizio, come orari, ubicazioni e costi specifici. Si prevede di offrire circa 150 pacchetti turistici diversi.

Le prenotazioni effettuate dai clienti rappresentano un altro aspetto centrale del progetto. Ogni prenotazione sarà univocamente identificata e associata sia a un cliente che a un pacchetto. Sarà necessario registrare informazioni quali la data della prenotazione, il numero di partecipanti, lo stato della prenotazione (ad esempio confermata, in attesa o annullata) e i dettagli relativi ai pagamenti.

La gestione dei pagamenti dovrà essere altrettanto accurata. Ogni pagamento sarà identificato da un codice univoco e la base di dati dovrà registrare l'importo, la data, il metodo di pagamento (come carta di credito o bonifico) e il riferimento alla prenotazione associata. Sarà importante supportare sia pagamenti completi che pagamenti parziali o rateizzati. Si prevede di elaborare circa 7.000 transazioni di pagamento annuali.

Infine, la base di dati dovrà includere informazioni sui fornitori esterni con cui l'agenzia collabora. Ogni fornitore sarà identificato da un codice univoco e potrà appartenere a diverse categorie, come compagnie aeree o hotel. Per ciascun fornitore sarà necessario memorizzare il nome, i contatti (telefono ed email), l'indirizzo. L'agenzia prevede di gestire circa 150 fornitori.

Analisi delle Specifiche

L'analisi delle specifiche consiste nel raccogliere e organizzare le informazioni necessarie al funzionamento della base di dati. Si procede estrapolando le frasi generali e più importanti dividendole in macroargomenti.

Frasi di carattere generale:

- L'agenzia di viaggi organizza e vende pacchetti turistici che possono includere voli, soggiorni in hotel, escursioni,e altre attività.
- L'agenzia collabora con fornitori esterni (compagnie aeree, catene alberghiere, tour operator) per offrire i suoi servizi.
- La base di dati deve tenere traccia di tutti i clienti, delle loro prenotazioni e dei dettagli relativi ai viaggi.

Frasi relative ai clienti:

- Ogni cliente è identificato da un codice univoco.
- Per ogni cliente sono memorizzati nome, cognome, data di nascita, contatti (telefono, email) e indirizzo.
- È possibile registrare le preferenze di viaggio di un cliente (es. tipo di destinazioni preferite, budget, tipologia di servizi richiesti).
- Ogni cliente può avere uno storico delle prenotazioni effettuate.

Frasi relative ai pacchetti:

- Ogni pacchetto è identificato da un codice univoco.
- Per ogni viaggio sono memorizzati il nome del pacchetto, la descrizione, la durata, le date di partenza e ritorno, la destinazione principale e il prezzo.
- Un pacchetto può includere più servizi: voli, soggiorni in hotel, escursioni, ecc.
- Ogni servizio all'interno di un pacchetto è descritto da dettagli come orari, luogo e costi specifici.

Frasi relative alle prenotazioni:

- Ogni prenotazione è identificata da un codice univoco e associata a un cliente e a uno o più viaggi.

- Per ogni prenotazione si memorizzano la data di prenotazione, il numero di partecipanti, lo stato della prenotazione (es. confermata, annullata) e i dettagli sui pagamenti.
- Ogni prenotazione può prevedere richieste speciali da parte del cliente (es. preferenze di budget e clima).

Frasi relative ai pagamenti:

- Ogni pagamento è identificato da un codice univoco.
- Per ogni pagamento si registra l'importo, la data, il metodo di pagamento (es. carta di credito, bonifico) e il riferimento alla prenotazione associata.
- La base di dati deve supportare pagamenti parziali (rate) o completi.

Frasi relative ai fornitori:

- Ogni fornitore è identificato da un codice univoco e può appartenere a diverse categorie (compagnia aerea, catena alberghiera, ecc.).
- Per ogni fornitore si memorizzano nome, contatti (telefono, email), indirizzo e condizioni contrattuali (es. sconti, termini di pagamento).
- I fornitori offrono i servizi utilizzati nei viaggi organizzati dall'agenzia.

Frasi relative allo staff:

- Ogni dipendente è identificato da un codice univoco.
- Per ogni dipendente si registra nome, cognome, ruolo (es. assistenza clienti, amministratore), contatti e storico delle prenotazioni gestite.
- Gli agenti di viaggio possono essere associati a uno o più clienti per offrire un servizio personalizzato.

Frasi relative alle destinazioni:

- Ogni destinazione è identificata da un codice univoco.
- Per ogni destinazione si memorizzano il nome, il paese, la città, la descrizione e le attrazioni principali.

Frasi relative alle recensioni:

- I clienti possono lasciare recensioni sui viaggi effettuati.
- Ogni recensione è identificata da un codice univoco e associata a un viaggio e a un cliente.
- Per ogni recensione si registra la data e una valutazione in stelle (da 1 a 5).

Glossario dei termini principali

TERMINE	DESCRIZIONE	SINONIMI	TERMINI COLLEGATI
Cliente	Persona che usufruisce dei servizi offerti dall'agenzia.		Prenotazione, storico prenotazioni, preferenze di viaggio, pagamento
Pacchetto	Pacchetto turistico organizzato dall'agenzia che include vari servizi (voli, hotel o escursioni).	Viaggio	Servizi, destinazione, durata, fornitori
Recensione	Opinione del cliente su un viaggio o servizio, che può essere positiva o negativa	Valutazione	Cliente, viaggio, pacchetto
Pagamento	Transazione finanziaria legata a una prenotazione, che può essere completa, parziale o rateizzata.		Prenotazione, importo, metodo di pagamento, stato
Prenotazione	Atto di riservare un viaggio, associato a un cliente e a uno o più servizi, con tutti i dettagli.		Cliente, pacchetto, stato, pagamento
Fornitore	Azienda esterna che collabora con l'agenzia per fornire servizi come voli, hotel o escursione.	Fornitori esterni	Pacchetto, servizi, contatti, sconto
Dipendenti	Persone che lavorano per l'agenzia e che contribuiscono alla gestione dell'attività.	Staff, Agenti di viaggio	Agenzia, fornitori, clienti, servizi

Elenco delle operazioni

L'elenco delle operazioni principali per la base di dati completa la fase preliminare della progettazione concettuale. L'informazione e la frequenza con cui sono svolte tali operazioni diventano determinanti in fase di progettazione logica in termini di ottimizzazione dell'informazione da rappresentare.

OP1: Aggiunta di un nuovo cliente
Registrazione di un nuovo cliente nel sistema. (Frequenza: 1-3 volte al giorno)

OP2: Modifica delle informazioni di un cliente
Aggiornamento di dati esistenti per un cliente. (Frequenza: 1-2 volte al giorno)

OP3: Visualizzazione dello storico prenotazioni di un cliente
Visualizzazione delle prenotazioni passate di un cliente con descrizione sommaria. (Frequenza: 1 volta al mese)

OP4: Eliminazione di un cliente
Rimozione del profilo cliente dal sistema, incluse tutte le prenotazioni associate.
(Frequenza: 1 volta al mese)

OP5: Creazione di un nuovo pacchetto
Creazione di un nuovo pacchetto turistico con attributi specifici.
(Frequenza: 1 volta a settimana)

OP6: Modifica dei dettagli di un pacchetto
Aggiornamento dei dettagli di un viaggio esistente. (Frequenza: 2-3 volte al mese)

OP7: Visualizzazione dei pacchetti disponibili in base a destinazione e prezzo
Consultazione dei pacchetti turistici in base ai criteri selezionati.
(Frequenza: 10 volte al giorno)

OP8: Visualizzazione dei pacchetti disponibili in base al clima estivo o invernale della Destinazione Principale
Consultazione dei pacchetti turistici in base ai criteri selezionati.
(Frequenza: 5-15 volte al giorno)

OP9: Cancellazione di un pacchetto
Rimozione di un viaggio o pacchetto turistico dal sistema. (Frequenza: 1-2 volte al mese)

OP10: Creazione di una nuova prenotazione
Registrazione di una nuova prenotazione da parte di un cliente.
(Frequenza: 10 volte al giorno)

OP11: Modifica di una prenotazione esistente
Aggiornamento di dettagli di una prenotazione, come date o partecipanti.
(Frequenza: 3-5 volte al giorno)

OP12: Visualizzazione dello stato di una prenotazione
Controllo dello stato di una prenotazione. (Frequenza: 10-20 volte al giorno)

OP13: Annullamento di una prenotazione
Cancellazione di una prenotazione e aggiornamento dello stato.
(Frequenza: 3-5 volte alla settimana)

OP14: Registrazione di un nuovo pagamento
Registrazione di un pagamento effettuato per una prenotazione.
(Frequenza: 5-15 volte al giorno)

OP15: Modifica dei dettagli di un pagamento
Aggiornamento dei dati relativi a un pagamento già effettuato.
(Frequenza: 1-2 volte al giorno)

OP16: Visualizzazione dei pagamenti associati a una prenotazione
Consultazione dei pagamenti effettuati per una prenotazione specifica.
(Frequenza: 5-10 volte al giorno)

OP17: Annullamento di un pagamento
Cancellazione di un pagamento e aggiornamento dello stato. (Frequenza: 2 volte a settimana)

OP18: Aggiunta di un nuovo fornitore
Registrazione di un nuovo fornitore nel sistema. (Frequenza: 1-2 volte al mese)

OP19: Modifica delle informazioni di un fornitore
Aggiornamento dei dati relativi a un fornitore. (Frequenza: 2-5 volte al mese)

OP20: Visualizzazione dei fornitori associati ad un pacchetto
Consultazione dei fornitori associati a un determinato viaggio.
(Frequenza: 3-10 volte al giorno)

OP21: Eliminazione di un fornitore
Rimozione di un fornitore dal sistema. (Frequenza: 1 volta al mese)

OP24: Visualizzazione delle destinazioni associate ai pacchetti
Consultazione delle destinazioni incluse nei viaggi. (Frequenza: 5-15 volte al giorno)

OP25: Inserimento di una nuova recensione da parte di un cliente
Registrazione di una recensione lasciata da un cliente per un viaggio.
(Frequenza: 1-5 volte al giorno)

OP26: Visualizzazione delle recensioni per un pacchetto
Consultazione delle recensioni lasciate da altri clienti per un viaggio.
(Frequenza: 5-10 volte al giorno)

OP27: Eliminazione di una recensione
Rimozione di una recensione dal sistema.
(Frequenza: 1 volta al mese)

OP28: Inserimento di un servizio ad un pacchetto
Aggiunta di un nuovo servizio ad un pacchetto; fatto per la creazione di ogni pacchetto, 6 servizi in media a pacchetto.
(Frequenza: 6 volte a settimana)

OP29: Visualizzazione del costo del pacchetto
Stampa il costo del pacchetto desiderato.
(Frequenza: 10 volte al giorno)

OP30: Aggiornamento del prezzo dei servizi
Modifica del costo di un servizio; il pacchetto rimane invariato come numero e tipologia di servizi, ma può aumentare o diminuire di prezzo a seconda del costo dei servizi stessi.
(Frequenza: 10 volta a settimana)

OP31: Eliminazione di un servizio
Cancellazione di un servizio dal database.
(Frequenza: 1 volta al mese)

Schema Entity-Relationship

Il punto di partenza per la realizzazione del modello concettuale sono i tre concetti cardine ovvero Cliente, Pacchetto, Staff:

- Cliente identifica tutti i clienti che usufruiscono dei servizi offerti dall'agenzia di viaggi.
- Pacchetto rappresenta un particolare viaggio che l'agenzia fornisce.
- Staff identifica i vari dipendenti che lavorano all'interno dell'agenzia.

Lo schema scheletro iniziale è quindi rappresentato dalle tre entità CLIENTE, PACCHETTO e STAFF legate tra loro dalla relationship Prenotazione le cui occorrenze sono una terna di occorrenze delle tre entità coinvolte: rappresentano la prenotazione di un pacchetto, effettuata da ogni cliente, e il relativo dipendente dell'agenzia che gestirà la prenotazione.

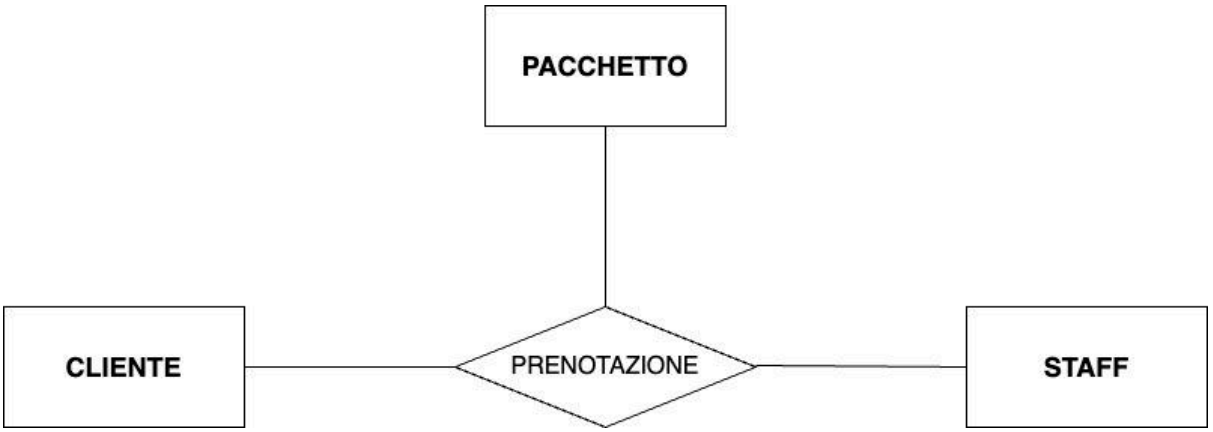

Figura 1: Schema scheletro (versione 1)

Questo schema non permette di rappresentare correttamente tutte le possibili relazioni tra le tre entità, si procede perciò alla raffinazione dei concetti tramite strategia di progetto mista.

In particolare si effettueranno inizialmente una primitiva di trasformazione top-down e due bottom-up: una rettificazione di una relazione, cioè la trasformazione della stessa in un'entità, l'introduzione di nuove entità e l'introduzione di nuove relazioni.
È necessario ciò per prevedere la presenza di più prenotazioni da parte dello stesso cliente con lo stesso dipendente dell'agenzia.

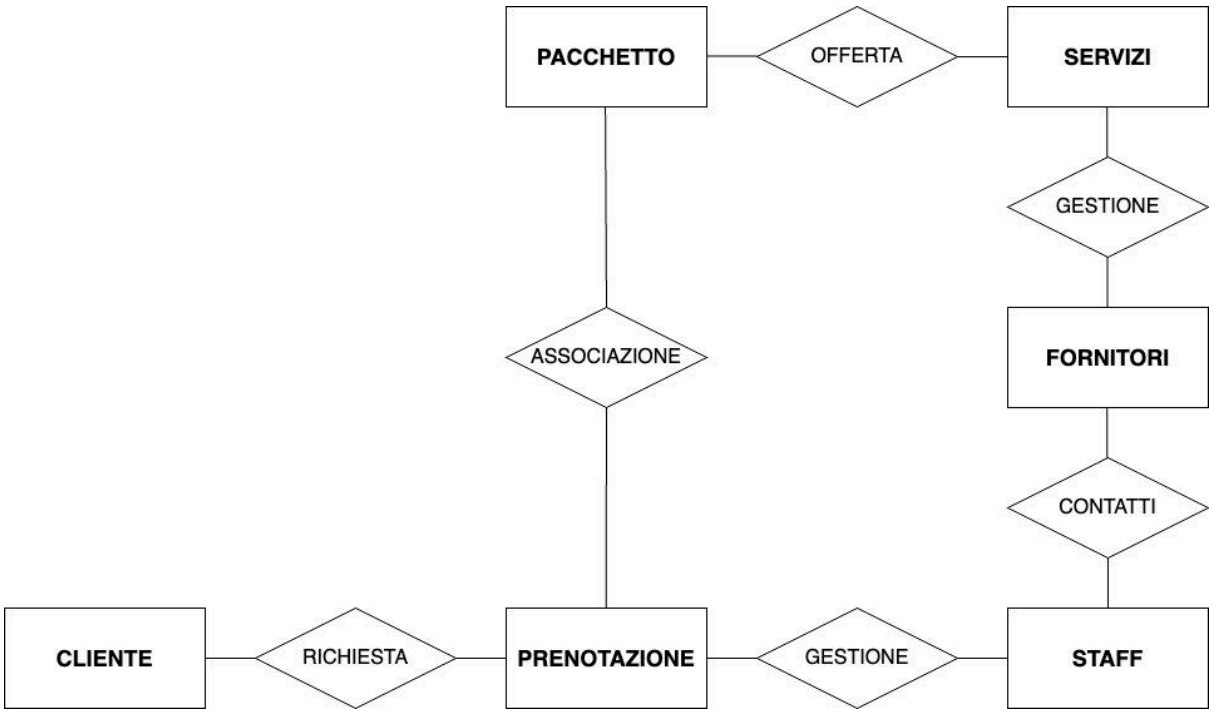

Figura 2: Evoluzione schema (versione 2)

La figura 2 ci mostra il secondo step della nostra costruzione, in particolare, abbiamo aggiunto l'entità SERVIZI molto importante per completare delle informazioni inerenti a pacchetto e FORNITORI che "forniscono" i servizi. Questa nuova entità è in relazione con PACCHETTO e FORNITORI tramite le relazioni OFFERTA e GESTIONE. Possiamo notare che questa versione anche se leggermente più descrittiva manca sia di risoluzione nel descrivere i comportamenti tra le maggiori Entità sia di generalizzazioni di FORNITORI e anche di STAFF i quali sono ancora concettualmente non completi.

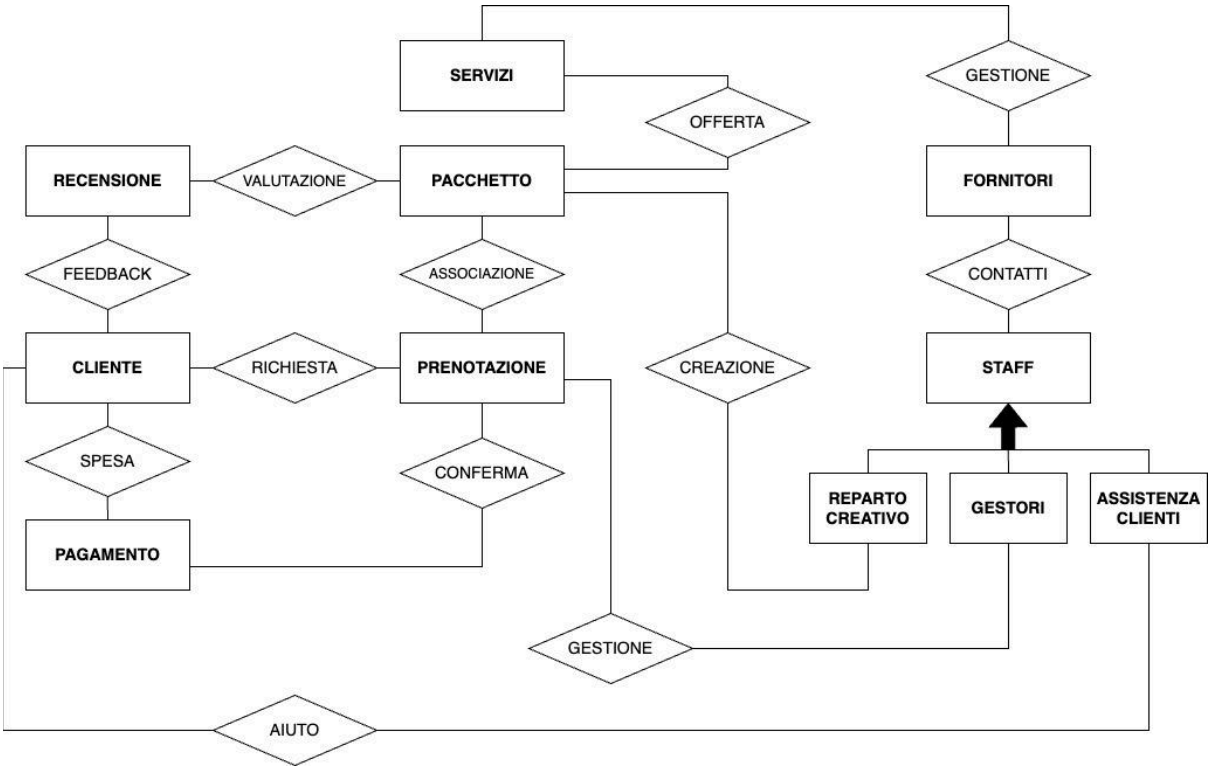

Figura 3: Evoluzione schema (versione 3)

La versione sovrastante risolve parte degli svantaggi della versione precedente. L'entità STAFF è diventata la generalizzazione di (REPARTO CREATIVO, GESTORI e ASSISTENZA CLIENTI) i quali hanno 3 relazioni differenti : (CLIENTE con REPARTO CREATIVO, PRENOTAZIONE con GESTORI e PACCHETTO con ASSISTENZA CLIENTI), con questa generalizzazioni si sottolineano le differenti funzioni del personale. Nella figura 3 possiamo anche notare l'inclusione di due entità: RECENSIONE che si occuperà di gestire le informazioni relative alle recensioni per ogni pacchetto date le recensioni lasciate dai clienti. Un altra entità aggiunta è PAGAMENTO che ha relazione con PRENOTAZIONE e CLIENTE. Questa entità si occupa di tener traccia di pagamenti dei clienti verso le prenotazioni fatte, verrà ampliata in seguito.

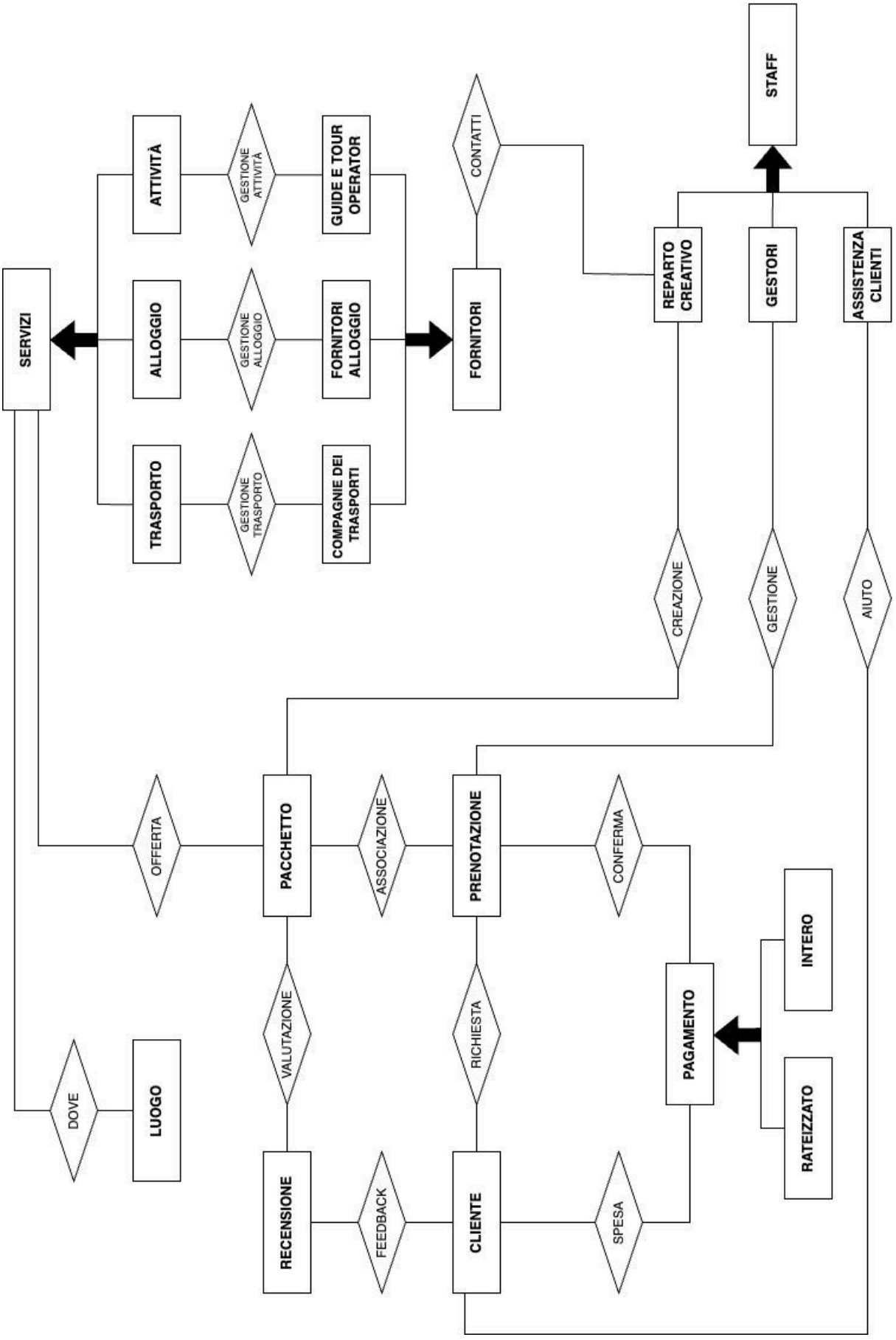

Figura 4: Evoluzione schema (versione 4)

Nella quarta versione dello schema concettuale è stata introdotta l'entità LUOGO, che organizza le informazioni relative alle destinazioni dei viaggi e alle sedi di erogazione dei servizi associati. L'entità FORNITORI diventa una generalizzazione che comprende le sottoclassi (COMPAGNIE DEI TRASPORTI, FORNITORI ALLOGGIO e GUIDE E TOUR OPERATOR). Ogni sottoclasse di FORNITORE è in relazione con una delle 3 sottoclassi di SERVIZI che in questa versione è diventata una generalizzazione. Le relazioni sono rispettivamente: (TRASPORTI e COMPAGNIE DI TRASPORTO), (ALLOGGIO e FORNITORI ALLOGGIO), (ATTIVITÀ e GUIDE E TOUR OPERATOR). La relazione PAGAMENTO è diventata generalizzazione di RATEIZZATO e INTERO. Questa generalizzazione servirà per distinguere le informazioni di un pagamento rateizzato e quello intero che avranno degli attributi differenti.

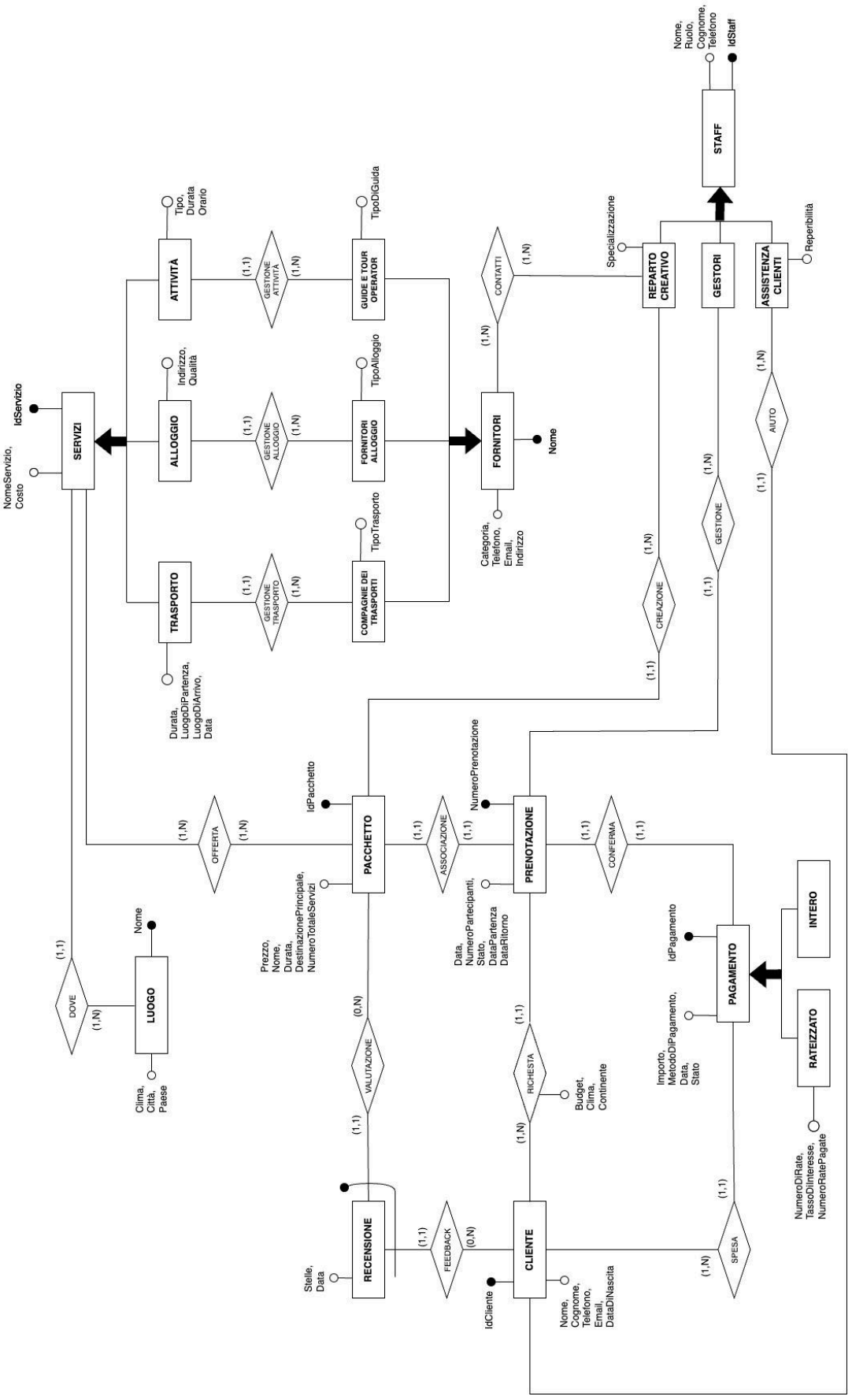

Figura 5: Schema completo

Vincoli non esprimibili

- Un cliente non può prenotare due pacchetti diversi per la stessa data.
- La disponibilità e il prezzo del servizio varia a seconda della stagione e della domanda.
- Un cliente non può prenotare un pacchetto che include un'attività per cui non ha l'età minima richiesta.
- Il numero di partecipanti a un'escursione non può superare la capacità massima del mezzo di trasporto

Dizionario dei dati (Entità)

ENTITA'	IDENTIFICATORE	ATTRIBUTI	DESCRIZIONE
Cliente	IdCliente	Nome, Cognome, Telefono, Email, DataDiNascita	Persona che si rivolge all'agenzia
Staff	IdStaff	Nome, Ruolo, Cognome, Telefono	Insieme dei dipendenti dell'agenzia
Reparto Creativo	IdStaff	Specializzazione	Dipendenti che si occupano di ideare nuovi pacchetti
Assistenza	IdStaff	Reperibilità	Dipendenti che forniscono assistenza ai clienti
Gestori	IdStaff		Dipendenti che si occupano di gestire le prenotazioni
Prenotazione	NumeroPrenotazione	Data, NumeroPartecipanti, Stato, DataPartenza, DataRitorno	Realizzazione della richiesta effettuata dal cliente
Recensione	IdCliente, IdPacchetto	Stelle, Data	Opinione del cliente sul pacchetto acquistato
Pacchetto	IdPacchetto	Prezzo, Nome, Durata, Destinazione Principale, Numero Totale di Servizi	Prodotto offerto dall'agenzia

Servizi	IdServizio	Costo, NomeServizio	Servizi inclusi nel pacchetto
Alloggio	IdServizio	Indirizzo, Qualità	Servizi dedicati agli alloggi
Transporto	IdServizio	Durata, LuogoDiPartenza, LuogoDiArrivo, Data	Servizi dedicati ai trasporti
Attività	IdServizio	Durata, Tipo, Descrizione	Servizi dedicati alle attività
Fornitore	Nome	Categoria, Telefono, Email, Indirizzo	Insieme degli enti che si occupano di fornire servizi
Compagnie di Trasporto	Nome	TipoTrasporto	Fornitori di trasporti
Fornitori Alloggio	Nome	TipoAlloggio	Fornitori di alloggi
Guide e Tour Operator	Nome	TipoGuida	Fornitori di attività
Pagamento	IdPagamento	Importo, Data, MetodoDiPagamento, Stato	Informazioni sui pagamenti
Intero	IdPagamento		Pagamento effettuato in un unicum
Rateizzato	IdPagamento	NumeroDiRate, TassoDiInteresse, NumeroRatePagate	Pagamento effettuato a rate
Luogo	Città, Paese	Clima	Luogo del viaggio

Dizionario dei dati (Relazioni)

RELAZIONE	ENTITA’	ATTRIBUTI	DESCRIZIONE
Richiesta	Cliente, Prenotazione	Budget, Clima, Continente	Storico delle prenotazioni effettuate da ciascun cliente

FeedBack	Cliente, Recensione		Recensione del pacchetto da parte del cliente
Valutazione	Recensione, Pacchetto		Valutazione del pacchetto a cui si riferisce la recensione
Associazione	Pacchetto, Prenotazione		Associazione di un pacchetto alla relativa prenotazione
Conferma	Prenotazione, Pagamento		Conferma della prenotazione tramite pagamento
Spesa	Cliente, Pagamento		Prezzo del pacchetto che il cliente deve pagare
Dove	Servizi, Luogo		Associazione della destinazione ai servizi del pacchetto
Contatti	Staff, Fornitori		Richiesta di servizi ai fornitori esterni da parte dello staff
Gestione Trasporti	Compagnie dei trasporti, Trasporti		Gestione dei mezzi di trasporto forniti dalla compagnia
Gestione Alloggi	Fornitori di alloggi, Alloggi		Gestione degli alloggi forniti dall'agenzia
Gestione Attività	Guide e Tour Operator, Attività		Gestione delle attività fornite da guide e tour operator
Gestione	Gestori, Prenotazione		Gestione della prenotazione da parte del dipendente gestore
Creazione	Reparto creativo, Pacchetto		Creazione del pacchetto da parte del dipendente del reparto creativo
Aiuto	Assistenza, Clienti		Assistenza da parte dell'agenzia al cliente richiedente aiuto

PROGETTAZIONE LOGICA

In questa fase, al fine di creare uno schema logico consono per una traduzione diretta in schema relazionale, è necessario riorganizzare e modificare lo schema E–R ottenuto durante la progettazione concettuale.

Tavola dei volumi

CONCETTO	TIPO	VOLUME
Clienti	E	5000
Prenotazioni	E	1.5 x 5000 (clienti)= 7500
Staff	E	20
Reparto Creativo	E	8
Assistenza	E	3
Gestori	E	9
Fornitori	E	50
Compagnie di trasporti	E	10
Fornitori di alloggi	E	20
Guide e Tour Operator	E	20
Servizi	E	500
Attività	E	100
Trasporto	E	200
Alloggio	E	200
Luogo	E	100
Recensione	E	6000
Pacchetto	E	150
Pagamento	E	10000
Intero	E	4000
Rateizzato	E	6000

Richiesta	R	7500
Spesa	R	10000
Associazione	R	7500
Valutazione	R	6000
Feedback	R	6000
Offerta	R	150
Contatti	R	22500
Gestione Trasporti	R	7500
Gestione Alloggi	R	7500
Gestione Attività	R	7500
Gestione	R	15000
Dove	R	100
Creazione	R	150
Aiuto	R	5500
Conferma	R	7000

Tavola delle operazioni

OPERAZIONE	TIPO	Frequenza
OP1	I	1-3 volte al giorno
OP2	I	1-2 volte al giorno
OP3	I	1 volta a settimana
OP4	B	1 volta al mese
OP5	B	1 volta a settimana
OP6	B	2-3 volte al mese
OP7	I	5-15 volte al giorno
OP8	I	5-15 volte al giorno
OP9	B	1-2 volte al mese
OP10	I	10 volte al giorno
OP11	I	3-5 volte al giorno
OP12	I	10-20 volte al giorno
OP13	I	3-5 volte a settimana

OP14	I	5-15 volte al giorno
OP15	I	1-2 volte al giorno
OP16	I	5-10 volte al giorno
OP17	B	1-2 volte al mese
OP18	B	1-2 volte al mese
OP19	B	2-5 volte al mese
OP20	I	3-10 volte al giorno
OP21	B	1 volta al mese
OP22	I	5 volte al giorno
OP23	I	3-5 volte al giorno
OP24	I	5-15 volte al giorno
OP25	I	1-5 volte al giorno
OP26	I	5-10 volte al giorno
OP27	B	1 volta al mese
OP28	B	50 volte a settimana
OP29	I	10 volte a settimana
OP30	B	1 volta a settimana
OP31	B	1 volta al mese

Analisi delle Ridondanze

Una possibile ridondanza riguarda la presenza dell'attributo Prezzo dell'entità Pacchetto. Questo valore infatti può essere calcolato dinamicamente sommando i costi dei servizi di cui un pacchetto è composto, contenuti nelle occorrenze della relazione Offerta.

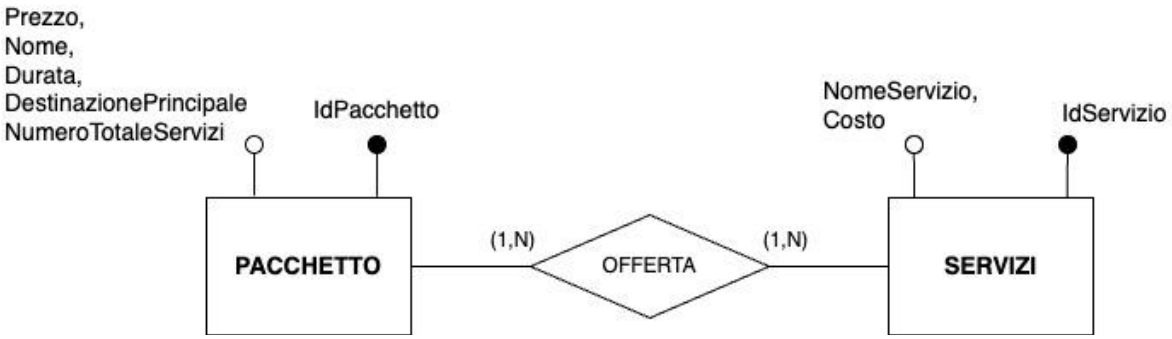

Figura 6: Schema con ridondanza

Per determinare se conviene o meno mantenere questa ridondanza occorre analizzare e quindi confrontare gli indici di prestazione nei due casi, con e senza ridondanza.
L'indice di prestazione viene calcolato sulla base delle operazioni che coinvolgono relazioni ed entità, oggetto della ridondanza.

In questo caso specifico, le operazioni di cui vanno valutati gli accessi sono:

OP28: Visualizzazione del costo del pacchetto
Frequenza: 10 volte al giorno (50 volte a settimana).

OP29: Aggiornamento del costo dei servizi offerti dai pacchetti
(Il pacchetto rimane invariato come numero e tipologia di servizi, ma può aumentare o diminuire di prezzo a seconda del costo dei servizi stessi. Occorre ricalcolare il prezzo per tutti i pacchetti che includono quel servizio). Frequenza: 10 volte a settimana.

Si consideri che **l'OP28** viene effettuata ogni volta che un cliente fa una richiesta di prenotazione (Frequenza: 10 volte al giorno).

Ogni pacchetto offre in media 6 servizi e uno stesso servizio si trova in media in 2 pacchetti diversi.

Cominciamo con l'analizzare gli accessi in presenza di ridondanza, cioè quando è presente in Pacchetto l'attributo Prezzo.

Tavola degli accessi per **OP28**:

CONCETTO	COSTRUTTO	ACCESSI	TIPO
Pacchetto	E	1	Lettura

L'operazione necessita di un solo accesso in lettura per cercare il pacchetto di interesse di cui si vuole sapere il prezzo.

Il costo in termini di accessi è: 1L*50 = **50 accessi a settimana**

Tavola degli accessi per **OP29**:

CONCETTO	COSTRUTTO	ACCESSI	TIPO
Servizi	E	1	Lettura
Servizi	E	1	Scrittura
Offerta	R	2	Lettura
Offerta	R	2	Scrittura
Pacchetto	E	2	Lettura
Pacchetto	E	2	Scrittura

L'operazione richiede:

- un accesso in lettura all'entità Servizi per cercare il servizio di interesse e un accesso in scrittura per modificare il suo costo.
- due accessi in lettura alla relazione Offerta perché in media uno stesso servizio è offerto da due pacchetti distinti e due accessi in scrittura per modificare il valore dell'attributo costo nelle due occorrenze.
- due accessi in lettura all'entità Pacchetto perché in media due pacchetti contengono lo stesso servizio e due accessi in scrittura per modificare il loro prezzo.

Considerando il peso di una scrittura pari a 2 letture e moltiplicando i risultati parziali per la frequenza dell'operazione, otteniamo che il costo a settimana in termini di accessi, in presenza di ridondanza, è dato da:

[(1L + 1S) + (2L + 2S) + (2L + 2S)]*f = (5S*2 + 5L)*f con **f = 10** (accessi a settimana)

= 15*10 = **150 accessi a settimana**

TOT: In presenza di ridondanza si hanno **200 accessi a settimana**.

Valutiamo ora i costi in assenza di ridondanza. Lo schema rimane lo stesso, salvo non essere più presente l'attributo Prezzo.

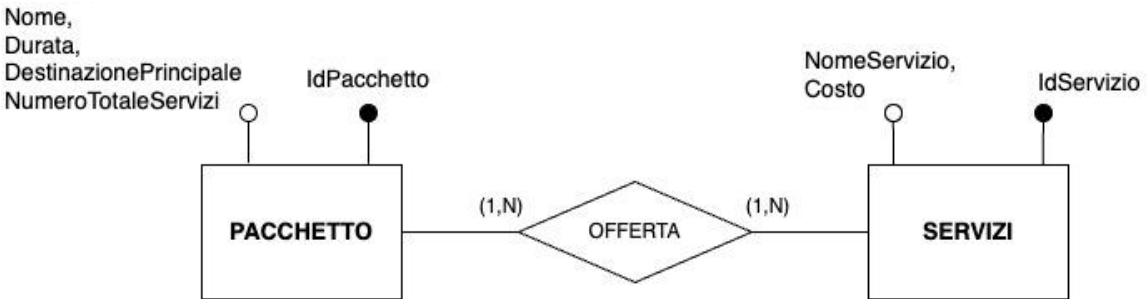

Figura 7: Schema senza ridondanza

Tavola degli accessi per **OP28:**

CONCETTO	COSTRUTTO	ACCESSI	TIPO
Pacchetto	E	1	Lettura
Offerta	R	6	Lettura

Il costo in assenza di ridondanza è: 7L*50 = **350 accessi a settimana**

Tavola degli accessi per **OP29**:

CONCETTO	COSTRUTTO	ACCESSI	TIPO
Servizi	E	1	Lettura
Servizi	E	1	Scrittura
Offerta	R	2	Lettura
Offerta	R	2	Scrittura

Il costo in assenza di ridondanza è:

[(1S + 1L) + (2L + 2S)]*10 = (3S*2 +3L)*10 = 9*10 = **90 accessi a settimana**

TOT: In assenza di ridondanza si hanno **440 accessi a settimana.**

Conclusione: risulta evidente che, in assenza di ridondanza, il numero di accessi risulti maggiore rispetto al caso con ridondanza. Abbiamo dunque deciso di mantenere il dato ridondante, nonché l'attributo Prezzo in Pacchetto.

Eliminazione delle Generalizzazione

Lo schema concettuale presenta quattro generalizzazioni totali, riguardanti le entità Staff, Fornitori, Servizi e Pagamento.

Analizzando le prime due generalizzazioni possiamo procedere **eliminando l'entità padre**, **mantenendo le entità figlie** e **aggiungendo delle nuove relazioni**, poiché nelle operazioni è di interesse distinguere solo tra le loro entità figlie.

GENERALIZZAZIONE DI FORNITORI

La seguente immagine mostra l'entità Fornitore prima e dopo la ristrutturazione:

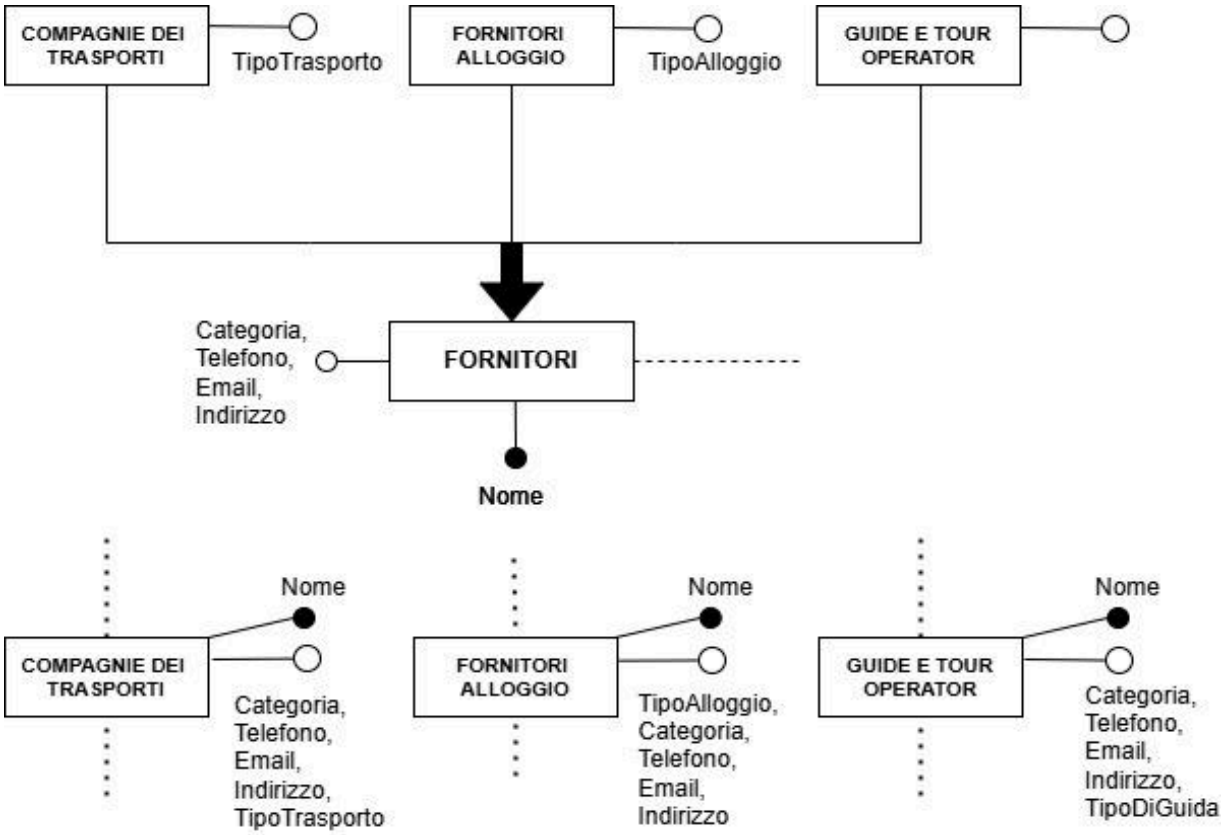

Figura 8: Generalizzazione di Fornitori con relativa ristrutturazione

GENERALIZZAZIONE DI STAFF

La seguente immagine mostra l'entità Staff prima e dopo la ristrutturazione:

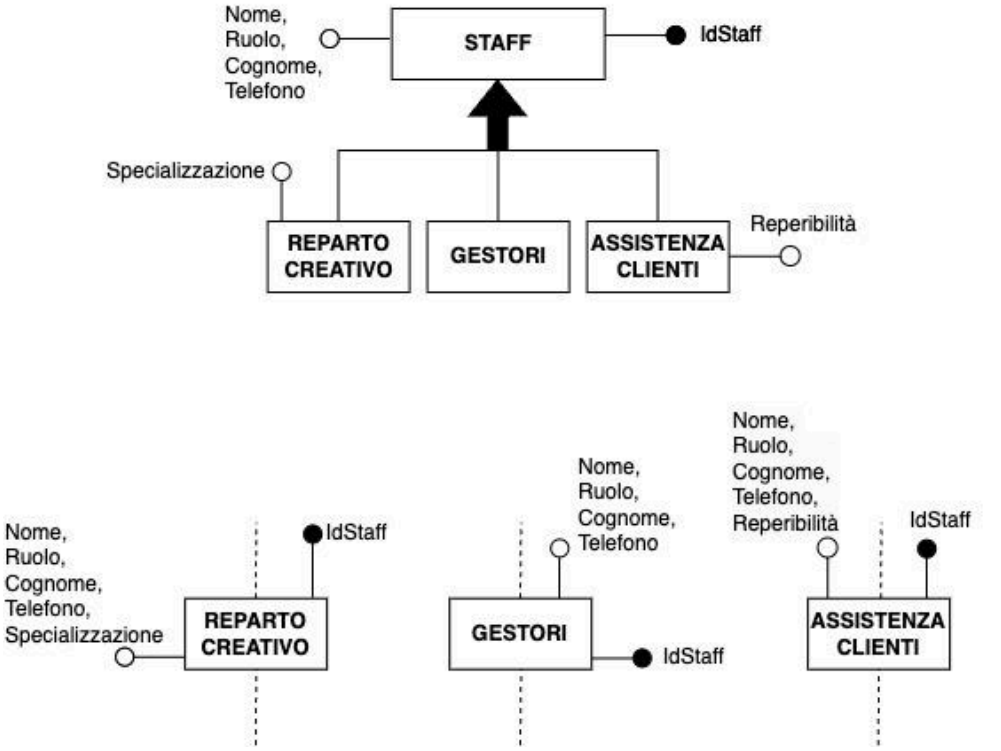

Figura 9: Generalizzazione di Staff con relativa ristrutturazione

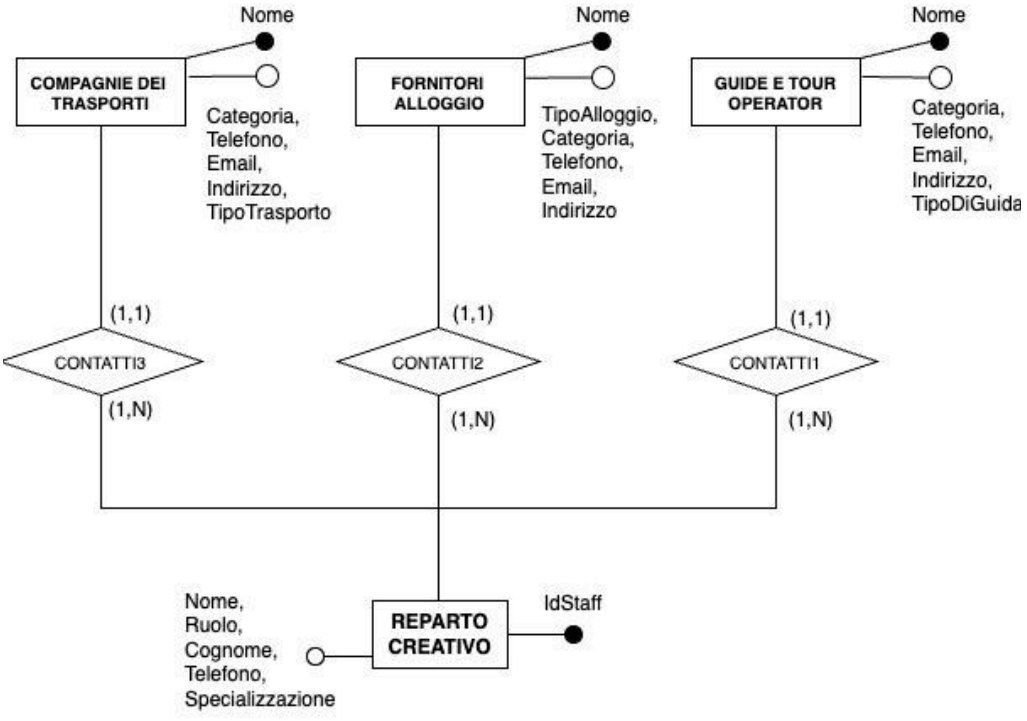

Figura 10: Generalizzazione di Staff

In Figura 10 viene mostrato che, dopo la ristrutturazione della generalizzazione Fornitori, sono state create tre nuove relazioni *Contatti* tra l'entità Reparto Creativo e le tre entità che prima erano specializzazione di Fornitori.

GENERALIZZAZIONE DI PAGAMENTO

Per quanto riguarda, invece, la generalizzazione in Pagamento, si è deciso di accorpare le entità figlie Rateizzato e Intero, nell'unica entità padre Pagamento, aggiungendo a quest'ultima anche gli attributi dell'entità Rateizzato. Infatti, un pagamento Intero può essere considerato come un pagamento con una sola rata e tasso di interesse pari a 0.

La figura sottostante mostra il risultato:

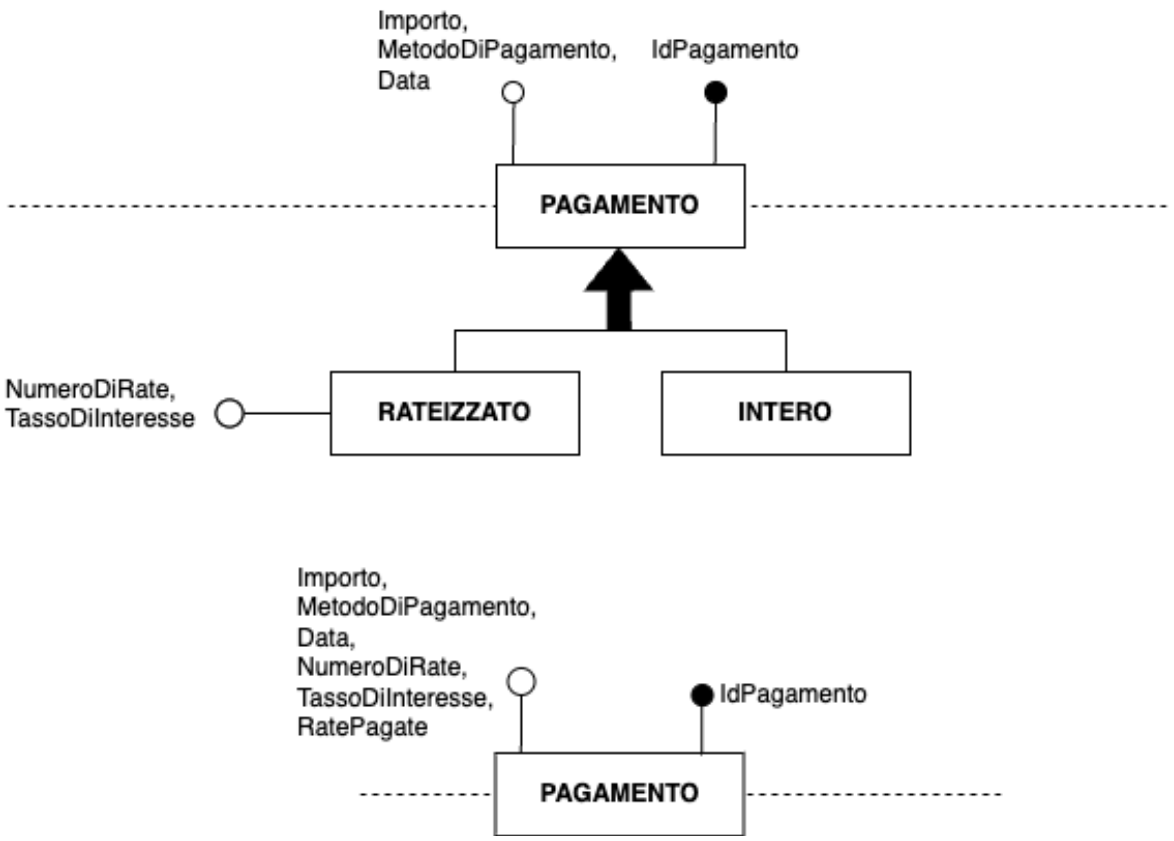

Figura 11: Generalizzazione di Pagamento con relativa ristrutturazione

GENERALIZZAZIONE DI SERVIZI

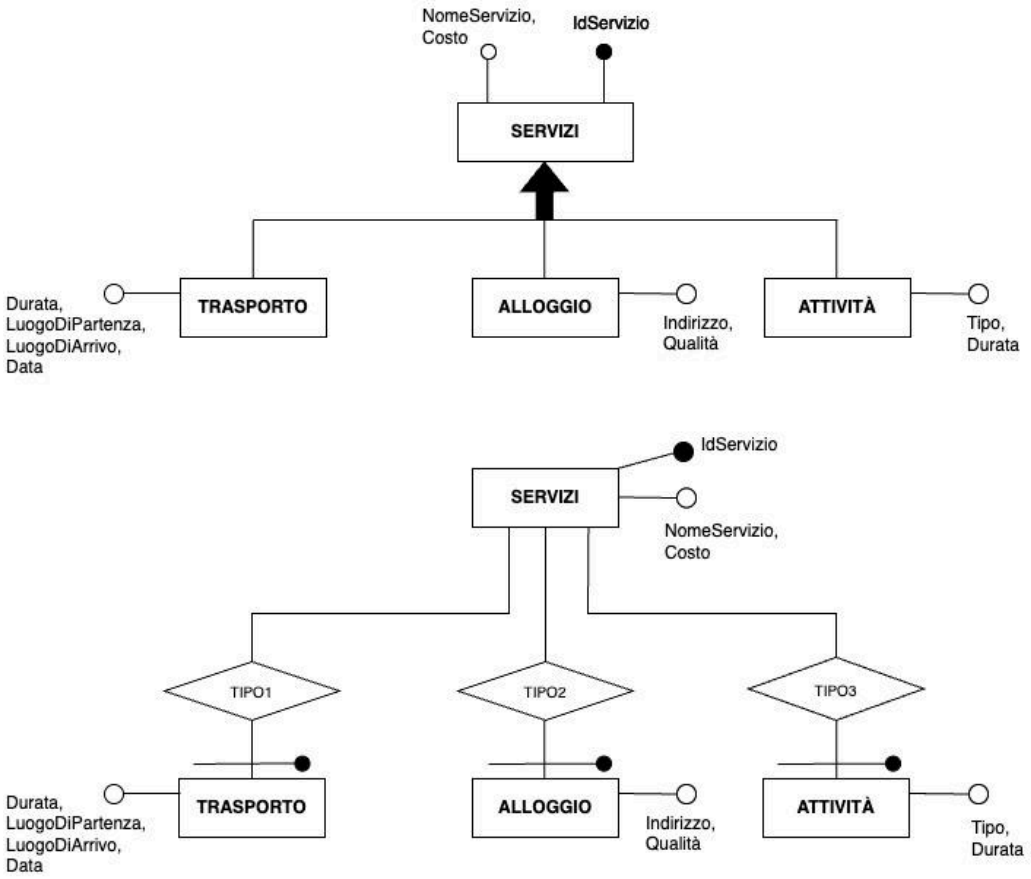

Figura 12: Generalizzazione di Servizi con relativa ristrutturazione

Infine, per quanto riguarda la generalizzazione in Servizi, non è possibile rimuovere né le entità padre né figlie, in quanto entrambe sono importanti per l'organizzazione della base di dati. Si è deciso piuttosto di creare 3 nuove relazioni *Tipo* e di aggiungere un identificatore esterno per ciascuna identità figlia.

Accorpamento/partizionamento di relationship

Lo schema non presenta possibili relationship/entità da accorpare o partizionare. Non sono presenti attributi multivalore.

Scelta degli identificatori primari

Abbiamo optato come scelta degli identificatori primari un codice alfanumerico progressivo per quasi ogni relazione. Il nome di questo attributo è formato così: (Id + il nome della relazione), la scelta di questo identificatore ci permette di avere una chiave di 1 solo attributo evitando di renderla più complessa tramite la formazione con più attributi.

Ogni relazione avrà però un dominio della chiave differente affinché si potranno utilizzare codici per ogni relazione senza creare conflitti. Questi domini possono variare nella dimensione :((A11 è diverso da 0A11, il quale ha un campo di 4 simboli) e nella disposizione delle lettere e dei numeri utilizzata. Per un numero minore di relazione si è deciso di utilizzare tipicamente il Nome come in FORNITORE.

La decisione di utilizzare principalmente un codice progressivo serve per semplificare il costo di aggiunta di una tupla, non utilizzando funzioni di hash per generarne. Lo svantaggio principale di questa scelta sta nel non poter usufruire dei codici progressivi che vengono cancellati alla cancellazione di una tupla. Per questo nella progettazioni si dovrà gestire tramite un'ottimizzazione l'utilizzo di questi codici i quali potrebbero saturare.

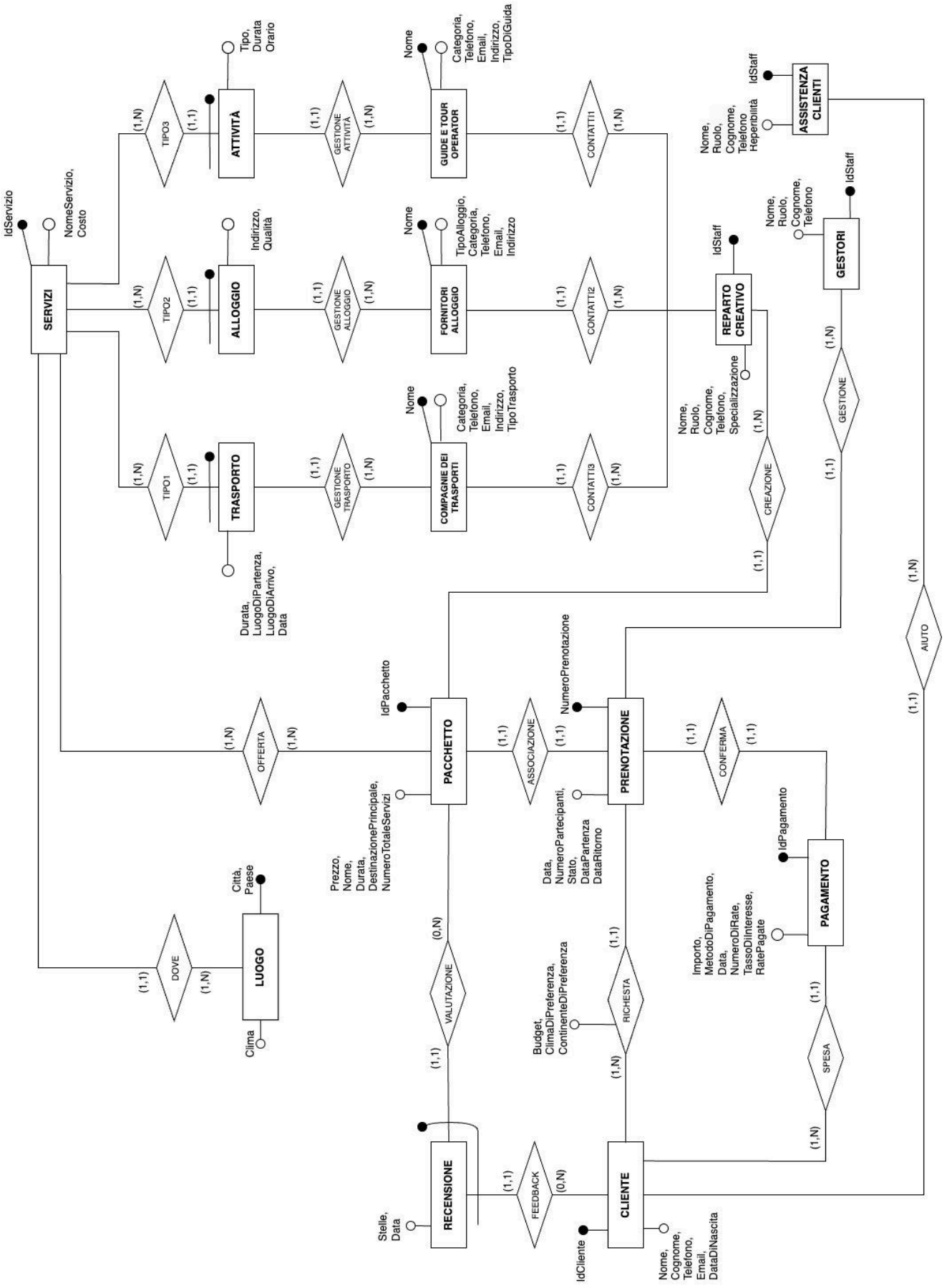

Figura 13: Schema finale ristrutturato

Modello Relazionale

A partire dallo schema E/R ristrutturato si costruisce uno schema logico equivalente in grado di rappresentare le medesime informazioni. Tale schema, chiamato modello relazione, costituisce il risultato della progettazione logica. Procediamo nell'elencare l' Entità e le Relazioni del nostro schema.

Traduzione di Entità:

CLIENTE (IdCliente, Nome, Cognome, NumeroTelefono, Email, DataDiNascita).

ASSISTENZACLIENTI (IdStaff, Reperibilità, Nome, Cognome, Ruolo, Telefono).

GESTORI (IdStaff, Nome, Cognome, Ruolo, Telefono).

PACCHETTO (IdPacchetto, Prezzo, Nome, Durata, DestinazionePrincipale, NumeroTotaleServizi).

ALLOGGIO (IdServizio, Nome, Costo, Indirizzo, Qualità).

- Vincolo di integrità referenziale tra l'attributo IdServizio e la relazione Servizi.

TRASPORTO (IdServizio, Nome, Costo, Data, Durata, LuogoDiPartenza, LuogoDiArrivo).

- Vincolo di integrità referenziale tra l'attributo IdServizio e la relazione Servizi.

ATTIVITÀ (IdServizio, Nome, Costo, Tipo, Durata, Orario).

- Vincolo di integrità referenziale tra l'attributo IdServizio e la relazione Servizi.

RECENSIONE (IdCliente, IdPacchetto, Stelle, Data).

- Vincolo di integrità referenziale tra l'attributo IdCliente e la relazione Cliente.
- Vincolo di integrità referenziale tra l'attributo IdPacchetto e la relazione Pacchetto.

PAGAMENTO (IdPagamento, Importo, Data, MetodoPagamento, NumeroDiRate, TassoDiInteresse, RatePagate).

LUOGO (Città, Paese, Clima).

PRENOTAZIONE (NumeroPrenotazione, Data, NumeroPartecipanti, Stato, DataPartenza, DataRitorno).

COMPAGNIEDEITRASPORTI (Nome, Categoria, Telefono, Email, Indirizzo, TipoTrasporto).

FORNITORIALLOGGIO (Nome, TipoAlloggio, Categoria, Telefono, Email, Indirizzo).

GUIDETOUROPERATOR (Nome, Categoria, Telefono, Email, Indirizzo, TipoDiGuida).

REPARTOCREATIVO (IdStaff, Specializzazione, Nome, Ruolo, Cognome, Telefono).

SERVIZI (IdServizio, NomeServizio, Costo).

Traduzione di Relazioni:

RICHIESTA(Cliente, Prenotazione, Budget, Clima, Continente)

- Vincolo di integrità referenziale fra l'attributo Cliente di RICHIESTA e la relazione CLIENTE.
- Vincolo di integrità referenziale fra l'attributo Prenotazione di RICHIESTA e la relazione PRENOTAZIONE.

FEEDBACK(Cliente, Recensione)

- Vincolo di integrità referenziale fra l'attributo Cliente di FEEDBACK e la relazione CLIENTE.
- Vincolo di integrità referenziale tra l'attributo Recensione di FEEDBACK e la relazione RECENSIONE.

VALUTAZIONE(Recensione, Pacchetto)

- Vincolo di integrità referenziale tra l'attributo Recensione di VALUTAZIONE e la relazione RECENSIONE.
- Vincolo di integrità referenziale fra l'attributo Pacchetto di VALUTAZIONE e la relazione PACCHETTO.

ASSOCIAZIONE(Pacchetto, Prenotazione)

- Vincolo di integrità referenziale tra l'attributo Pacchetto di ASSOCIAZIONE e la relazione PACCHETTO.
- Vincolo di integrità referenziale tra l'attributo Prenotazione di ASSOCIAZIONE e la relazione PRENOTAZIONE.

CONFERMA(Prenotazione, Pagamento)

- Vincolo di integrità referenziale fra l'attributo Prenotazione di CONFERMA e la relazione PRENOTAZIONE.
- Vincolo di integrità referenziale tra l'attributo Pagamento di CONFERMA e la relazione PAGAMENTO.

SPESA(Cliente, Pagamento)

- Vincolo di integrità referenziale tra l'attributo Cliente di SPESA e la relazione CLIENTE.
- Vincolo di integrità referenziale fra l'attributo Pagamento di SPESA e la relazione PAGAMENTO.

OFFERTA(Pacchetto, Servizi)

- Vincolo di integrità referenziale tra l'attributo Pacchetto di OFFERTA e la relazione PACCHETTO.
- Vincolo di integrità referenziale tra l'attributo Servizi di OFFERTA e la relazione SERVIZI.

DOVE(Servizi, Destinazione)

- Vincolo di integrità referenziale tra l'attributo Servizi di DOVE e la relazione SERVIZI.
- Vincolo di integrità referenziale tra l'attributo Destinazione di DOVE e la relazione LUOGO.

CONTATTI1 (Staff, Fornitori)

- Vincolo di integrità referenziale tra l'attributo Staff di CONTATTI1 e la relazione REPARTOCREATIVO.
- Vincolo di integrità referenziale tra l'attributo Fornitori di CONTATTI1 e la relazione COMPAGNIEDEITRASPORTI.

CONTATTI2 (Staff, Fornitori)

- Vincolo di integrità referenziale tra l'attributo Staff di CONTATTI2 e la relazione REPARTOCREATIVO.
- Vincolo di integrità referenziale tra l'attributo Fornitori di CONTATTI2 e la relazione FORNITORIALLOGGIO.

CONTATTI3 (Staff, Fornitori)

- Vincolo di integrità referenziale tra l'attributo Staff di CONTATTI3 e la relazione REPARTOCREATIVO.
- Vincolo di integrità referenziale tra l'attributo Fornitori di CONTATTI3 e la relazione GUIDEETOUROPERATOR.

GESTIONETRASPORTO (CompagnieTrasporto, Trasporto)

- Vincolo di integrità referenziale tra l'attributo CompagnieTrasporto di GESTIONETRASPORTO e la relazione COMPAGNIEDEITRASPORTI.
- Vincolo di integrità referenziale fra l'attributo Trasporto di GESTIONETRASPORTO e la relazione TRASPORTO.

GESTIONEALLOGGIO (FornitoriAlloggio, Alloggio)

- Vincolo di integrità referenziale tra l'attributo FornitoriAlloggio di GESTIONEALLOGGIO e la relazione FORNITORIALLOGGIO.
- Vincolo di integrità referenziale tra l'attributo Alloggio di GESTIONEALLOGGIO e la relazione ALLOGGIO.

GESTIONEATTIVITA (GuideTourOperator, Attività)

- Vincolo di integrità referenziale tra l'attributo GuideTourOperator di GESTIONEATTIVITA e la relazione GUIDETOUROPERATOR.

- Vincolo di integrità referenziale tra l'attributo Attività di GESTIONEATTIVITA e la relazione ATTIVITÀ.

GESTIONE (Gestori, Prenotazione)

- Vincolo di integrità referenziale tra l'attributo Gestori di GESTIONE e la relazione GESTORI.
- Vincolo di integrità referenziale tra l'attributo Prenotazione di GESTIONE e la relazione PRENOTAZIONE.

CREAZIONE (RepartoCreativo, Pacchetto)

- Vincolo di integrità referenziale tra l'attributo RepartoCreativo di CREAZIONE e la relazione REPARTOCREATIVO.
- Vincolo di integrità referenziale tra l'attributo Pacchetto di CREAZIONE e la relazione PACCHETTO.

AIUTO (Assistenza, Cliente)

- Vincolo di integrità referenziale tra l'attributo Assistenza di AIUTO e la relazione ASSISTENZACLIENTI.
- Vincolo di integrità referenziale tra l'attributo Cliente di AIUTO e la relazione CLIENTE.

SPESA (Cliente, Pagamento)

- Vincolo di integrità referenziale tra l'attributo Cliente di SPESA e la relazione CLIENTE.
- Vincolo di integrità referenziale tra l'attributo Pagamento di SPESA e la relazione PAGAMENTO.

OFFERTA (Pacchetto, Servizi)

- Vincolo di integrità referenziale tra l'attributo Pacchetto di OFFERTA e la relazione PACCHETTO.
- Vincolo di integrità referenziale tra l'attributo Servizi di OFFERTA e la relazione SERVIZI.

TIPO1 (Servizi, Trasporto)

- Vincolo di integrità referenziale tra l'attributo Servizi di TIPO1 e la relazione SERVIZI.
- Vincolo di integrità referenziale tra l'attributo Trasporto di TIPO1 e la relazione TRASPORTO.

TIPO2 (Servizi, Alloggio)

- Vincolo di integrità referenziale tra l'attributo Servizi di TIPO2 e la relazione SERVIZI.

- Vincolo di integrità referenziale tra l'attributo Alloggio di TIPO2 e la relazione ALLOGGIO.

TIPO3 (Servizi, Attività)

- Vincolo di integrità referenziale tra l'attributo Servizi di TIPO3 e la relazione SERVIZI.
- Vincolo di integrità referenziale tra l'attributo Attività di TIPO3 e la relazione ATTIVITÀ.

IMPLEMENTAZIONE DELLE OPERAZIONI

Creazione delle tabelle

Creazione della tabella **CLIENTE**:

```
CREATE TABLE Cliente (
      IdCliente CHAR(10) PRIMARY KEY,
      Nome VARCHAR(50) NOT NULL,
      Cognome VARCHAR(50) NOT NULL,
      NumeroTelefono VARCHAR(15),
      Email VARCHAR(100),
      DataDiNascita DATE
);
```

Creazione della tabella **ASSISTENZA**:

```
CREATE TABLE Assistenza (
    IdStaff CHAR(10) PRIMARY KEY,
    Nome VARCHAR(50) NOT NULL,
    Cognome VARCHAR(50) NOT NULL,
    Ruolo VARCHAR(50),
    Reperibilità VARCHAR(50),
    Telefono VARCHAR(15)
);
```

Creazione della tabella **GESTORI**:

```
CREATE TABLE Gestori (
    IdStaff CHAR(10) PRIMARY KEY,
    Nome VARCHAR(50) NOT NULL,
    Cognome VARCHAR(50) NOT NULL,
    Ruolo VARCHAR(50),
    Telefono VARCHAR(15)
);
```

Creazione della tabella **REPARTO CREATIVO:**

```
CREATE TABLE RepartoCreativo (
    IdStaff CHAR(10) PRIMARY KEY,
    Specializzazione VARCHAR(100)
);
```

Creazione della tabella **PACCHETTO**:

```
CREATE TABLE Pacchetto (
    IdPacchetto CHAR(10) PRIMARY KEY,
    Nome VARCHAR(100) NOT NULL,
    Destinazione VARCHAR(100) NOT NULL,
    Durata INT NOT NULL,
    Prezzo DECIMAL(10, 2) NOT NULL
);
```

Creazione della tabella **ALLOGGIO**:

```
CREATE TABLE Alloggio (
    IdServizio CHAR(10) PRIMARY KEY,
    Nome VARCHAR(100),
    Costo DECIMAL(10, 2),
    Indirizzo VARCHAR(200),
    Qualità VARCHAR(50)
);
```

Creazione della tabella **TRASPORTO**:

```
CREATE TABLE Trasporto (
    IdServizio CHAR(10) PRIMARY KEY,
    Nome VARCHAR(100),
    Costo DECIMAL(10, 2),
    Data DATE,
    Durata TIME,
    LuogoPartenza VARCHAR(100),
    LuogoArrivo VARCHAR(100)
);
```

Creazione della tabella **ATTIVITÀ**:

```
CREATE TABLE Attività (
    IdServizio CHAR(10) PRIMARY KEY,
    Nome VARCHAR(100),
    Durata TIME
);
```

Creazione della tabella **FORNITORE**:

```
CREATE TABLE Fornitore (
    Nome VARCHAR(100) PRIMARY KEY,
    Categoria VARCHAR(50),
    Telefono VARCHAR(15),
    Email VARCHAR(100),
    Indirizzo VARCHAR(200)
);
```

Creazione della tabella **PRENOTAZIONE**:

```
CREATE TABLE Prenotazione (
    NumeroPrenotazione CHAR(10) PRIMARY KEY,
    Data DATE NOT NULL,
    NumeroPartecipanti INT NOT NULL,
    Stato VARCHAR(50),
    DataPartenza DATE NOT NULL,
    DataRitorno DATE NOT NULL,
    IdCliente CHAR(10) REFERENCES Cliente(IdCliente)
        ON UPDATE CASCADE
        ON DELETE SET NULL,
    IdPacchetto CHAR(10) REFERENCES Pacchetto(IdPacchetto)
        ON UPDATE CASCADE
        ON DELETE SET NULL
);
```

Creazione della tabella **RECENSIONE**:

```
CREATE TABLE Recensione (

    IdCliente CHAR(10),

    IdPacchetto CHAR(10),

    Stelle INT CHECK (Stelle BETWEEN 1 AND 5),

    Data DATE,

    PRIMARY KEY (IdCliente, IdPacchetto),

    FOREIGN KEY (IdCliente) REFERENCES Cliente(IdCliente)

        ON UPDATE CASCADE
        ON DELETE SET NULL,

    FOREIGN KEY (IdPacchetto) REFERENCES
    Pacchetto(IdPacchetto)

        ON UPDATE CASCADE
        ON DELETE CASCADE

);
```

Creazione della tabella **PAGAMENTO**:

```
CREATE TABLE Pagamento (

    IdPagamento CHAR(10) PRIMARY KEY,

    Importo DECIMAL(10, 2) NOT NULL,

    Data DATE NOT NULL,

    MetodoPagamento VARCHAR(50),

    NumeroRate INT DEFAULT 1,

    TassoDiInteresse DECIMAL(5, 2) DEFAULT 0,

    ImportoPerRata DECIMAL(10, 2),

    ImportoGiàVersato DECIMAL(10, 2),

    NumeroPrenotazione CHAR(10),
```

```
    FOREIGN KEY (NumeroPrenotazione)    REFERENCES
    Prenotazione(NumeroPrenotazione)
                                        ON UPDATE CASCADE
                                        ON DELETE SET NULL
);
```

Creazione della tabella **LUOGO**:

```
CREATE TABLE Luogo (
    Nome VARCHAR(100) PRIMARY KEY,
    Clima VARCHAR(50),
    Città VARCHAR(50),
    Paese VARCHAR(50)
);
```

Creazione della tabella **SERVIZI**:

```
CREATE TABLE Servizi (
    IdServizio CHAR(10) PRIMARY KEY,
    NomeServizio VARCHAR(100) NOT NULL,
    Costo DECIMAL(10, 2) NOT NULL
);
```

Creazione della tabella **COMPAGNIE DI TRASPORTO**:

```
CREATE TABLE CompagnieDeiTrasporti (
    Nome VARCHAR(100) PRIMARY KEY,
    TipoTrasporto VARCHAR(50)
);
```

Creazione della tabella **FORNITORI DI ALLOGGI**:

```
CREATE TABLE FornitoriAlloggio (
    Nome VARCHAR(100) PRIMARY KEY,
    TipoAlloggio VARCHAR(50)
);
```

Creazione della tabella **GUIDE E TOUR OPERATOR**:

```
CREATE TABLE GuideTourOperator (
    Nome VARCHAR(100) PRIMARY KEY,
    TipoDiGuida VARCHAR(50)
);
```

Creazione della relazione **OFFERTA** (Pacchetto - Servizi):

```
CREATE TABLE Offerta (
    IdPacchetto CHAR(10),
    IdServizio CHAR(10),
    PRIMARY KEY (IdPacchetto, IdServizio),
    FOREIGN KEY (IdPacchetto) REFERENCES
    Pacchetto(IdPacchetto)
                            ON UPDATE CASCADE
                            ON DELETE SET NULL,
     FOREIGN KEY (IdServizio) REFERENCES
     Servizi(IdServizio)
                            ON UPDATE CASCADE
                            ON DELETE SET NULL
);
```

Creazione della relazione **DOVE** (Servizi - Luogo):

```
CREATE TABLE Dove (

    IdServizio CHAR(10),

    NomeLuogo VARCHAR(100),

    PRIMARY KEY (IdServizio, NomeLuogo),

    FOREIGN KEY (IdServizio) REFERENCES Servizi(IdServizio),

          ON UPDATE CASCADE
          ON DELETE SET NULL,

    FOREIGN KEY (NomeLuogo) REFERENCES Luogo(Nome)

          ON UPDATE CASCADE
          ON DELETE SET NULL

);
```

Creazione della relazione **GESTIONE TRASPORTO** (Compagnie dei trasporti - Trasporto):

```
CREATE TABLE GestioneTrasporti (

    NomeCompagnia VARCHAR(100),

    IdServizio CHAR(10),

    PRIMARY KEY (NomeCompagnia, IdServizio),

    FOREIGN KEY (NomeCompagnia) REFERENCES
    CompagnieDeiTrasporti(Nome)

        ON UPDATE CASCADE
        ON DELETE SET NULL,

    FOREIGN KEY (IdServizio) REFERENCES Trasporto(IdServizio)

        ON UPDATE CASCADE
        ON DELETE SET NULL

);
```

Creazione della relazione **GESTIONE ALLOGGI** (Fornitori alloggio - Alloggio):

```
CREATE TABLE GestioneAlloggi (

    NomeFornitore VARCHAR(100),

    IdServizio CHAR(10),

    PRIMARY KEY (NomeFornitore, IdServizio),

    FOREIGN KEY (NomeFornitore)    REFERENCES
    FornitoriAlloggio(Nome)

          ON UPDATE CASCADE
          ON DELETE SET NULL,

    FOREIGN KEY (IdServizio) REFERENCES Alloggio(IdServizio)

          ON UPDATE CASCADE
          ON DELETE SET NULL

);
```

Creazione della relazione **GESTIONE ATTIVITÀ** (Guide e Tour Operator - Attività):

```
CREATE TABLE GestioneAttività (

    NomeGuida VARCHAR(100),

    IdServizio CHAR(10),

    PRIMARY KEY (NomeGuida, IdServizio),

    FOREIGN KEY (NomeGuida)   REFERENCES
    GuideTourOperator(Nome)
                              ON UPDATE CASCADE
                              ON DELETE SET NULL,

    FOREIGN KEY (IdServizio)  REFERENCES  Attività(IdServizio)
                              ON UPDATE CASCADE
                              ON DELETE SET NULL

);
```

Creazione della relazione **CREAZIONE PACCHETTI** (Reparto Creativo - Pacchetto):

```
CREATE TABLE Creazione (

    IdStaff CHAR(10),

    IdPacchetto CHAR(10),

    PRIMARY KEY (IdStaff, IdPacchetto),

    FOREIGN KEY (IdStaff) REFERENCES RepartoCreativo(IdStaff),

                              ON UPDATE CASCADE
                              ON DELETE SET NULL,

    FOREIGN KEY (IdPacchetto) REFERENCES
    Pacchetto(IdPacchetto)

                              ON UPDATE CASCADE
                              ON DELETE SET NULL

);
```

Creazione della relazione **GESTIONE PRENOTAZIONI** (Gestori - Prenotazione)

```
CREATE TABLE GestionePrenotazioni (

    IdStaff CHAR(10),

    NumeroPrenotazione CHAR(10),

    PRIMARY KEY (IdStaff, NumeroPrenotazione),

    FOREIGN KEY (IdStaff) REFERENCES Gestori(IdStaff),

    FOREIGN KEY (NumeroPrenotazione) REFERENCES
    Prenotazione(NumeroPrenotazione)
                                     ON UPDATE CASCADE
                                     ON DELETE SET NULL

);
```

Creazione della relazione **AIUTO** (Assistenza - Cliente):

```
CREATE TABLE Aiuto (

    IdStaff CHAR(10),

    IdCliente CHAR(10),

    PRIMARY KEY (IdStaff, IdCliente),

    FOREIGN KEY (IdStaff)     REFERENCES Assistenza(IdStaff)
                              ON UPDATE CASCADE
                              ON DELETE SET NULL,

    FOREIGN KEY (IdCliente)   REFERENCES Cliente(IdCliente)
                              ON UPDATE CASCADE
                              ON DELETE SET NULL

);
```

Creazione della relazione **FEEDBACK** (Cliente - Recensione):

```
CREATE TABLE Feedback (

    IdCliente CHAR(10),

    IdRecensione CHAR(10),

    PRIMARY KEY (IdCliente, IdRecensione),

    FOREIGN KEY (IdCliente)      REFERENCES Cliente(IdCliente)
                                 ON UPDATE CASCADE
                                 ON DELETE SET NULL,

    FOREIGN KEY (IdRecensione)
    REFERENCES Recensione(IdCliente, IdPacchetto)
                                 ON DELETE CASCADE
                                 ON UPDATE SET NULL,

);
```

Creazione delle query

OP1:

```
CREATE OR REPLACE FUNCTION AggiungiCliente(
    p_IdCliente CHAR(10),
    p_Nome VARCHAR(50),
    p_Cognome VARCHAR(50),
    p_NumeroTelefono VARCHAR(15),
    p_Email VARCHAR(100),
    p_DataDiNascita DATE
)
RETURNS VOID AS $$
BEGIN
    INSERT INTO Cliente (IdCliente, Nome, Cognome,
    NumeroTelefono, Email, DataDiNascita)
    VALUES (p_IdCliente, p_Nome, p_Cognome, p_NumeroTelefono,
    p_Email, p_DataDiNascita);
END;
$$ LANGUAGE plpgsql;
```

OP2:

```
CREATE OR REPLACE FUNCTION ModificaCliente(
    p_IdCliente CHAR(10),
    p_Nome VARCHAR(50),
    p_Cognome VARCHAR(50),
```

```
    p_NumeroTelefono VARCHAR(15)
)
RETURNS VOID AS $$
BEGIN
    UPDATE Cliente
    SET Nome = p_Nome, Cognome = p_Cognome, NumeroTelefono =
    p_NumeroTelefono
    WHERE IdCliente = p_IdCliente;
END;
$$ LANGUAGE plpgsql;
```

OP3:

```
CREATE OR REPLACE FUNCTION StoricoPrenotazioni(
    p_IdCliente CHAR(10)
)
RETURNS   TABLE(NumeroPrenotazione CHAR(10), Data DATE,
          NomePacchetto VARCHAR) AS $$
BEGIN
    RETURN QUERY
    SELECT P.NumeroPrenotazione, P.Data, Pa.Nome AS
    NomePacchetto
    FROM Prenotazione P JOIN Pacchetto Pa ON P.IdPacchetto =
    Pa.IdPacchetto
    WHERE P.IdCliente = p_IdCliente;
END;
$$ LANGUAGE plpgsql;
```

OP4:

```
CREATE OR REPLACE FUNCTION EliminaCliente(
    p_IdCliente CHAR(10)
)
RETURNS VOID AS $$
BEGIN
    DELETE FROM Prenotazione WHERE IdCliente = p_IdCliente;
    DELETE FROM Cliente WHERE IdCliente = p_IdCliente;
END;
$$ LANGUAGE plpgsql;
```

OP5:

```
CREATE OR REPLACE FUNCTION CreaPacchetto(
    p_IdPacchetto CHAR(10),
    p_Nome VARCHAR(100),
    p_Destinazione VARCHAR(100),
    p_Durata INT,
    p_Prezzo DECIMAL
)
RETURNS VOID AS $$
BEGIN
    INSERT INTO Pacchetto (IdPacchetto, Nome, Destinazione,
    Durata, Prezzo)
    VALUES (p_IdPacchetto, p_Nome, p_Destinazione, p_Durata,
    p_Prezzo);
END;
```

```
$$ LANGUAGE plpgsql;
```

OP6:

```
CREATE OR REPLACE FUNCTION ModificaPacchetto(
    p_IdPacchetto CHAR(10),
    p_Prezzo DECIMAL,
    p_Durata INT
)
RETURNS VOID AS $$
BEGIN
    UPDATE Pacchetto
    SET Prezzo = p_Prezzo, Durata = p_Durata
    WHERE IdPacchetto = p_IdPacchetto;
END;
$$ LANGUAGE plpgsql;
```

OP7:

```
CREATE OR REPLACE FUNCTION PacchettiPerDestinazionePrezzo(
    p_Destinazione VARCHAR(100),
    p_PrezzoMax DECIMAL
)
RETURNS TABLE(IdPacchetto CHAR(10), Nome VARCHAR(100), Prezzo
DECIMAL) AS $$
BEGIN
    RETURN QUERY
    SELECT IdPacchetto, Nome, Prezzo
```

```
    FROM Pacchetto

    WHERE Destinazione = p_Destinazione AND Prezzo <=
    p_PrezzoMax;

END;

$$ LANGUAGE plpgsql;
```

OP8:

```
CREATE OR REPLACE FUNCTION PacchettiPerClima(

    p_Clima VARCHAR(50)

)

RETURNS TABLE(IdPacchetto CHAR(10), Nome VARCHAR(100),
Destinazione VARCHAR(100)) AS $$

BEGIN

    RETURN QUERY

    SELECT P.IdPacchetto, P.Nome, P.Destinazione

    FROM Pacchetto P JOIN Luogo L ON P.Destinazione = L.Nome

    WHERE L.Clima = p_Clima;

END;

$$ LANGUAGE plpgsql;
```

OP9:

```
CREATE OR REPLACE FUNCTION EliminaPacchetto(

    p_IdPacchetto CHAR(10)

)

RETURNS VOID AS $$

BEGIN
```

```
    DELETE FROM Pacchetto WHERE IdPacchetto = p_IdPacchetto;
END;
$$ LANGUAGE plpgsql;
```

OP10:

```
CREATE OR REPLACE FUNCTION CreaPrenotazione(
    p_NumeroPrenotazione CHAR(10),
    p_Data DATE,
    p_NumeroPartecipanti INT,
    p_Stato VARCHAR(50),
    p_DataPartenza DATE,
    p_DataRitorno DATE,
    p_IdCliente CHAR(10),
    p_IdPacchetto CHAR(10)
)
RETURNS VOID AS $$
BEGIN
    INSERT INTO Prenotazione (NumeroPrenotazione, Data,
    NumeroPartecipanti, Stato, DataPartenza, DataRitorno,
    IdCliente, IdPacchetto)

    VALUES (p_NumeroPrenotazione, p_Data,p_NumeroPartecipanti,
    p_Stato, p_DataPartenza, p_DataRitorno, p_IdCliente,
    p_IdPacchetto);
END;
$$ LANGUAGE plpgsql;
```

OP11:

```
CREATE OR REPLACE FUNCTION ModificaPrenotazione(
    p_NumeroPrenotazione CHAR(10),
    p_NumeroPartecipanti INT,
    p_Stato VARCHAR(50)
)
RETURNS VOID AS $$
BEGIN
    UPDATE Prenotazione
    SET    NumeroPartecipanti = p_NumeroPartecipanti,
           Stato = p_Stato
    WHERE  NumeroPrenotazione = p_NumeroPrenotazione;
END;
$$ LANGUAGE plpgsql;
```

OP12:

```
CREATE OR REPLACE FUNCTION StatoPrenotazione(
    p_NumeroPrenotazione CHAR(10)
)
RETURNS TABLE(Stato VARCHAR(50)) AS $$
BEGIN
    RETURN QUERY
    SELECT Stato
    FROM Prenotazione
    WHERE NumeroPrenotazione = p_NumeroPrenotazione;
END;
```

```
$$ LANGUAGE plpgsql;
```

OP13:

```
CREATE OR REPLACE FUNCTION AnnullaPrenotazione(
    p_NumeroPrenotazione CHAR(10)
)
RETURNS VOID AS $$
BEGIN
    UPDATE Prenotazione
    SET Stato = 'Annullata'
    WHERE NumeroPrenotazione = p_NumeroPrenotazione;
END;
$$ LANGUAGE plpgsql;
```

OP14:

```
CREATE OR REPLACE FUNCTION RegistraPagamento(
    p_IdPagamento CHAR(10),
    p_Importo DECIMAL,
    p_Data DATE,
    p_MetodoPagamento VARCHAR(50),
    p_NumeroPrenotazione CHAR(10)
)
RETURNS VOID AS $$
BEGIN
```

```
    INSERT INTO Pagamento (IdPagamento, Importo, Data,
    MetodoPagamento, NumeroPrenotazione)

    VALUES (p_IdPagamento, p_Importo, p_Data,
    p_MetodoPagamento, p_NumeroPrenotazione);

END;

$$ LANGUAGE plpgsql;
```

OP15:

```
CREATE OR REPLACE FUNCTION ModificaPagamento(

    p_IdPagamento CHAR(10),

    p_MetodoPagamento VARCHAR(50)

)

RETURNS VOID AS $$

BEGIN

    UPDATE Pagamento

    SET MetodoPagamento = p_MetodoPagamento

    WHERE IdPagamento = p_IdPagamento;

END;

$$ LANGUAGE plpgsql;
```

OP16:

```
CREATE OR REPLACE FUNCTION PagamentiPerPrenotazione(

    p_NumeroPrenotazione CHAR(10)

)

RETURNS TABLE(IdPagamento CHAR(10), Importo DECIMAL, Data
DATE) AS $$

BEGIN
```

```
    RETURN QUERY
    SELECT IdPagamento, Importo, Data
    FROM Pagamento
    WHERE NumeroPrenotazione = p_NumeroPrenotazione;
END;
$$ LANGUAGE plpgsql;
```

OP17:

```
CREATE OR REPLACE FUNCTION AnnullaPagamento(
    p_IdPagamento CHAR(10)
)
RETURNS VOID AS $$
BEGIN
    DELETE FROM Pagamento WHERE IdPagamento = p_IdPagamento;
END;
$$ LANGUAGE plpgsql;
```

OP18:

```
CREATE OR REPLACE FUNCTION AggiungiFornitore(
    p_Nome VARCHAR(100),
    p_Categoria VARCHAR(50),
    p_Telefono VARCHAR(15),
    p_Email VARCHAR(100),
    p_Indirizzo VARCHAR(200)
)
RETURNS VOID AS $$
```

```
BEGIN
    INSERT INTO Fornitore (Nome, Categoria, Telefono, Email,
    Indirizzo)
    VALUES (p_Nome, p_Categoria, p_Telefono, p_Email,
    p_Indirizzo);
END;
$$ LANGUAGE plpgsql;
```

OP19:

```
CREATE OR REPLACE FUNCTION ModificaFornitore(
    p_Nome VARCHAR(100),
    p_Telefono VARCHAR(15),
    p_Email VARCHAR(100)
)
RETURNS VOID AS $$
BEGIN
    UPDATE Fornitore
    SET Telefono = p_Telefono, Email = p_Email
    WHERE Nome = p_Nome;
END;
$$ LANGUAGE plpgsql;
```

OP20:

```
CREATE OR REPLACE FUNCTION FornitoriPerPacchetto(
    p_IdPacchetto CHAR(10)
)
```

```
RETURNS TABLE(NomeFornitore VARCHAR(100), Categoria
VARCHAR(50)) AS $$

BEGIN

    RETURN QUERY

    SELECT F.Nome, F.Categoria

    FROM Fornitore F

    JOIN Offerta O ON F.Nome = O.IdServizio

    WHERE O.IdPacchetto = p_IdPacchetto;

END;

$$ LANGUAGE plpgsql;
```

OP21:

```
CREATE OR REPLACE FUNCTION EliminaFornitore(

    p_Nome VARCHAR(100)

)

RETURNS VOID AS $$

BEGIN

    DELETE FROM Fornitore WHERE Nome = p_Nome;

END;

$$ LANGUAGE plpgsql;
```

OP22:

```
CREATE OR REPLACE FUNCTION AggiungiRelazioneClientePacchetto(

    p_IdCliente CHAR(10),

    p_IdPacchetto CHAR(10)
```

```
)

RETURNS VOID AS $$

BEGIN

    INSERT INTO AssociazioneClientePacchetto (IdCliente,
    IdPacchetto)

    VALUES (p_IdCliente, p_IdPacchetto);

END;

$$ LANGUAGE plpgsql;
```

OP23:

```
CREATE OR REPLACE FUNCTION VisualizzaClientiPerPacchetto(

    p_IdPacchetto CHAR(10)

)

RETURNS TABLE(IdCliente CHAR(10), NomeCliente VARCHAR(50),
CognomeCliente VARCHAR(50)) AS $$

BEGIN

    RETURN QUERY

    SELECT C.IdCliente, C.Nome, C.Cognome

    FROM AssociazioneClientePacchetto ACP

    JOIN Cliente C ON ACP.IdCliente = C.IdCliente

    WHERE ACP.IdPacchetto = p_IdPacchetto;

END;

$$ LANGUAGE plpgsql;
```

OP24:

```
CREATE OR REPLACE FUNCTION DestinazioniAssociate()
RETURNS TABLE(Destinazione VARCHAR(100)) AS $$
BEGIN
    RETURN QUERY
    SELECT DISTINCT Destinazione
    FROM Pacchetto;
END;
$$ LANGUAGE plpgsql;
```

OP25:

```
CREATE OR REPLACE FUNCTION InserisciRecensione(
    p_IdCliente CHAR(10),
    p_IdPacchetto CHAR(10),
    p_Stelle INT,
    p_Data DATE
)
RETURNS VOID AS $$
BEGIN
    INSERT INTO Recensione (IdCliente, IdPacchetto, Stelle,
    Data)
    VALUES (p_IdCliente, p_IdPacchetto, p_Stelle, p_Data);
END;
$$ LANGUAGE plpgsql;
```

OP26:

```
CREATE OR REPLACE FUNCTION RecensioniPerPacchetto(
    p_IdPacchetto CHAR(10)
)
RETURNS TABLE(IdCliente CHAR(10), Stelle INT, Data DATE) AS $$
BEGIN
    RETURN QUERY
    SELECT IdCliente, Stelle, Data
    FROM Recensione
    WHERE IdPacchetto = p_IdPacchetto;
END;
$$ LANGUAGE plpgsql;
```

OP27:

```
CREATE OR REPLACE FUNCTION EliminaRecensione(
    p_IdCliente CHAR(10),
    p_IdPacchetto CHAR(10)
)
RETURNS VOID AS $$
BEGIN
    DELETE FROM Recensione
    WHERE IdCliente = p_IdCliente AND IdPacchetto =
    p_IdPacchetto;
END;
$$ LANGUAGE plpgsql;
```

OP28:

```
CREATE OR REPLACE FUNCTION AggiungiOfferta(
    p_IdPacchetto CHAR(10),
    p_IdServizio CHAR(10)
)
RETURNS VOID AS $$
BEGIN
    INSERT INTO Offerta (IdPacchetto, IdServizio)
    VALUES (p_IdPacchetto, p_IdServizio);
END;
$$ LANGUAGE plpgsql;
```

OP29:

```
CREATE OR REPLACE FUNCTION CostoPacchetto(
    p_IdPacchetto CHAR(10)
)
RETURNS DECIMAL AS $$
DECLARE
    costoTotale DECIMAL := 0;
BEGIN
    SELECT SUM(S.Costo)
    INTO costoTotale
    FROM Offerta O
    JOIN Servizi S ON O.IdServizio = S.IdServizio
    WHERE O.IdPacchetto = p_IdPacchetto;
```

```
    RETURN costoTotale;
END;
$$ LANGUAGE plpgsql;
```

OP30:

```
CREATE OR REPLACE FUNCTION AggiornaPrezzoServizio(
    p_IdServizio CHAR(10),
    p_NuovoCosto DECIMAL
)
RETURNS VOID AS $$
BEGIN
    UPDATE Servizi
    SET Costo = p_NuovoCosto
    WHERE IdServizio = p_IdServizio;
END;
$$ LANGUAGE plpgsql;
```

OP31:

```
CREATE OR REPLACE FUNCTION EliminaServizio(
    p_IdServizio CHAR(10)
)
RETURNS VOID AS $$
BEGIN
    DELETE FROM Offerta
    WHERE IdServizio = p_IdServizio;
```

```
    DELETE FROM Servizi
    WHERE IdServizio = p_IdServizio;
END;
$$ LANGUAGE plpgsql;
```