Elaborato Basi di Dati

Desiderio Edoardo

27 agosto 2023

Indice

Capitolo 1

Analisi dei requisiti

Lo scopo è realizzare un portale che permetta alla società richiedente di gestire in maniera informatizzata le prenotazini e l'organizzazione delle visite guidate che vuole organizzare con i siti di maggior interesse.

1.1 Intervista

Una società operante nel settore del turismo offre tra i suoi servizi l'organizzazione di visite guidate a siti d' interesse storico-culturale. Ogni visita, opportunamente descritta, ha un titolo (diverse visite hanno un titolo ricorrente, es. "Musei Vaticani e Cappella Sistina", "Sito archeologico di Pompei", "Galleria degli uffizi", ecc.), la sua durata media e il luogo in cui essa si svolge. Ogni visita può avere luogo più volte nel tempo secondo specifici eventi programmati. Le escursioni, di cui viene indicato il prezzo, vengono prenotati da gruppi di persone condotti da una guida che illustra il percorso in una determinata lingua; per ogni gruppo viene fissata l'ora d' inizio della visita e un numero minimo e massimo di partecipanti. Il prezzo degli eventi varia in base all'età:

- 0-12 il prezzo è gratuito
- 12-14 il prezzo è scontato del 20%
- gli over 50 godono di uno sconto pari al 10%

La società si avvale di diverse guide ognuna delle quali ha competenze in una o più lingue ad uno specifico livello di conoscenza ("B2","C1","C2"). Di ogni capo gruppo si vuole conoscere alcuni dati tra i quali nome, sesso, data di nascita, titolo di studio e relativo anno di conseguimento.
I clienti, di cui si vuole conoscere almeno nome, nazionalità, lingua base, e-mail

e un recapito telefonico, possono aggregarsi a uno o più gruppi, secondo le loro esigenze. Uno stesso visitatore, nel tempo, può partecipare a gruppi diversi usando ogni volta una certa forma di pagamento (non necessariamente sempre la stessa es. Carta di credito, paypal, bonifico bancario) della quale si deve prevedere la memorizzazione: tipologia, descrizione e data del pagamento. Il sito web della società consente la visione pubblica delle visite organizzate e, solo agli utenti preventivamente registrati, la prenotazione di una specifica visita. In fine l'applicativo deve permettere una visione protetta dei dati, quindi non tutti gli utenti ad esempio possono visionare i gruppi a cui sono affidate le guide

1.2 Rilevamento delle ambiguità e correzioni proposte

Il testo dell'intervista presenta molte ambiguità. Le principali sono

- utilizzo di sinonimi
- Elenchi di attributi incompleti
- Cartdinalità non specificate

Gli attributi parziali e le cardinalità verranno risolti mediante l'uso della logica in fase di creazione dello schema concettuale. Invece per quanto concerne i sinonimi, è necessario costruire un glossario dei termini

termine	descrizione	sinonimi	collegamenti
utente	entità che interagisce con il database lato consumatore	cliente, visitatore	gruppi, pagamenti, sconti
guida	figura qualificata in lingue e storia che illustra il percorso passo passo	capo gruppo, dipendente	competenze linguistiche, gruppi
sconto	rappresenta la percentuale da decurtare al prezzo finale in base all'età	-	pagamento, cliente
gruppo	insieme di persone in questo caso	-	cliente, guida, evento
evento	situazione specifica dato un luogo e orario	visita-guidata, escursioni	visita, gruppi
visite	logo d'interesse con cui ha accordi la società di turismo	sito culturale	eventi

Tabella 1.1:
termini rappresentativi dell' intervista

Ipotesi aggiuntive

dall'intervista fatta si concretizza che:

- il dato relativo alla durata media di una visita venga espresso in minuti
- per uno specifico evento di visita guidata possano essere formati anche più gruppi ognuno col proprio orario, accompagnatore e lingua;
- i vari visitatori per potersi iscriversi ad uno o più eventi debbono registrarsi sul sito della società fornendo e-mail e password. La banca

dati non prevede alcuna gestione relativamente agli utenti anonimi: essi possono operare solo per funzionalità limitate d' interrogazione per vedere i dati degli eventi programmati;

- per potersi iscrivere ad un gruppo di visita relativamente ad uno specifico evento, nei limiti della disponibilità di posti, ogni visitatore registrato effettui il pagamento tramite carta di credito (con codice della medesima), via PayPal (l'utente deve essere registrato a tale servizio), o tramite bonifico bancario di cui deve fornire gli estremi utilizzando il campo relativo alla descrizione del pagamento;
- il prezzo di una visita sia comunque individuale e venga espresso a livello di evento in quanto suscettibile di variazioni nel tempo
- per definire 'gratuito' il prezzo di un biglietto si imposterà una percentuale di sconto pari al 100%

1.3 Definizione delle specifiche in linguaggio naturale ed estrazione dei concetti principali

Di seguito si riporta un testo riassuntivo in cui sono evidenziati i concetti chiave dell' intervista filtrati dalle ambiguità possibili, in modo da avere un' idea più chiara di quelle che saranno le entità presenti nello schema concettuale.

La società commissionatrice vuole creare la gestione informatizzata dei suoi servizi.
Ogni **visita**, intesa come il sito culturale è opportunamente descritta definendo il titolo, luogo, identificativo, descrizione, durata media espressa in minuti. Per ogni visita la società organizza degli eventi. Ogni **evento** può ripetersi più volte rispetto ad una determinata visita e ne viene definito il prezzo indicato che può variare nel tempo; si vuole memorizzare anche la data dell'evento.

I **clienti**, di cui si salvano lingua preferenza e dati principali una password e mail per accedere a funzionalità più avanzate per la prenotazione di biglietti ecc, vengono assegnati a gruppi differenti in base alle loro preferenze proposte al momento dell'acquisto dei **biglietti**. Per gestire i pagamenti il cliente potrà aggiungere al carrello i biglietti che intende acquistare per poi procedere al pagamento utilizzando quello che più preferisce (paypall, bonifico, carta di credito)

Dei **Gruppi** si vuole indicare l'orario d' inizio della visita si vuole specificare il minimo numero di persone da cui deve essere composto un gruppo e il massimo, bisogna valutare se è conveniente salvare anche il numero d' iscritti correnti o lasciare il dato deducibile.

I **biglietti** devono essere acquistati dall'utente, siccome ogni utente può acquistare biglietti anche per altre persone ogni biglietto deve specificare il nome e il cognome e l'età della persona per cui viene acquistato il biglietto siccome sono previste fasce di **sconto** in base all'età. Rispettivamente $eta \leq 12 \rightarrow 100\%$, $12 < eta \leq 14 \rightarrow 20\%$, $eta > 50 \rightarrow 10\%$. Per ogni nuovo evento vengono emessi biglietti pari al numero minimo di partecipanti di un gruppo

IL **carrello** deve raccogliere le registrazioni per biglietti a persona per poi calcolare il totale dell'ordine. A livello applicativo sarà gestito da una vista che assocerà gli ordini fatti dal cliente

Delle **guide** turistiche oltre ai dati comuni con il cliente si vuole memorizzare anche il titolo di studio e il suo anno di conseguimento. La società intende registrare i dati relativi alle **competenze**. Ogni competenza deve essere riferita ad una guida e alle lingue che essa conosce, il livello di conoscenza deve essere espresso in (B2,C1,C2).

Individuazione operazioni principali

1. Aggiunta di visite
2. Aggiunta di eventi
3. inserimento di un nuovo cliente
4. inserimento di una nuova guida
5. supponendo che le guide si specializzino in altre lingue aggiunta di una nuova competenza di una guida
6. gestione e riepilogo ordini
7. Aggiunta di un nuovo gruppo
8. creazione di biglietti fino al numero minimo di partecipanti ad un gruppo
9. vendita di biglietti per un determinato cliente
10. applicazione sconto in base al destinatario del biglietto acquistato
11. storico degli acquisti
12. controllo del riempimento dei gruppi
13. assegnazione di una determinata guida ad un gruppo in base alle sue conoscenze
14. indicazione dei posti rimanenti per le iscrizioni ad un gruppo
15. aggiunta di una nuova lingua
16. inserimento di un nuovo metodo di pagamento
17. modifica valori Biglietto
18. creazione vista dello storico ordini per ogni cliente

19. Ricerca filtrata di annunci di visite da acquistare in base a diversi parametri scelti

20. ricerca delle visite disponibili in base ad una data

vedi stima frequenza piú avanti3.2

Capitolo 2

Progettazione concettuale

Per maggiore chiarezza, di seguito riportiamo i primi esempi di schemi concettuali divisi per ambiti. Partirò da parti di schema concettuale più semplici fino ad arrivare a parti più complesse

Visita-Evento-Gruppo

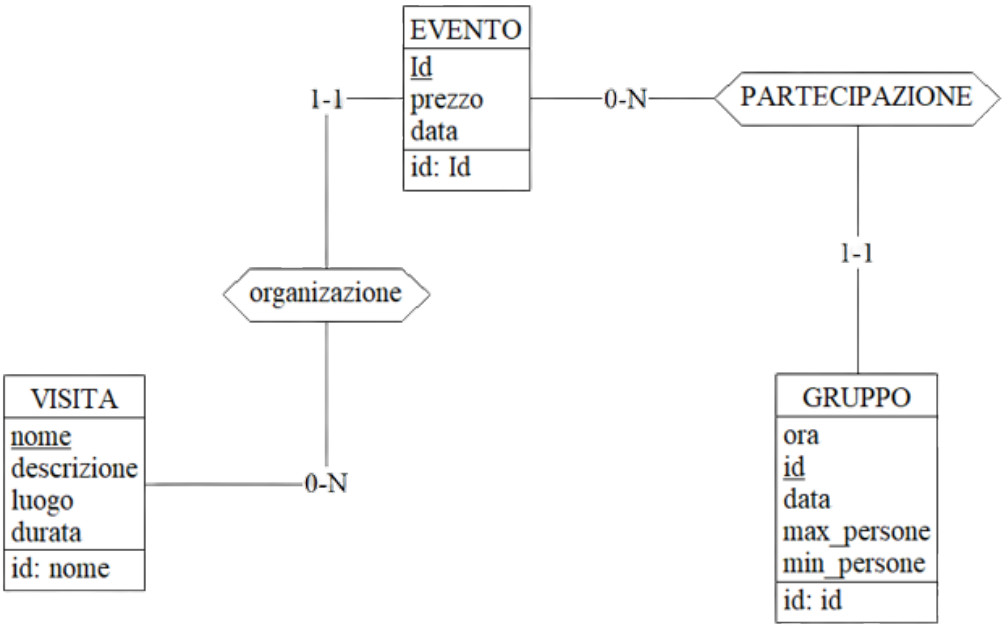


Figura 2.1: caratterizzazione della gestione fra le visite, i vari eventi che può ospitare e i gruppi che vi partecipano

Lo schema è piuttosto auto esplicativo, sono stati mantenuti i vincoli imposti dall'intervista come permettere l'Associazione di più eventi data una visita e la partecipazione di più gruppi al determinato evento si noti come da un gruppo si possa risalire ad un determinato evento e ad una determinata visita.

Gerarchia persone, relazione fra gruppi guide e competenze

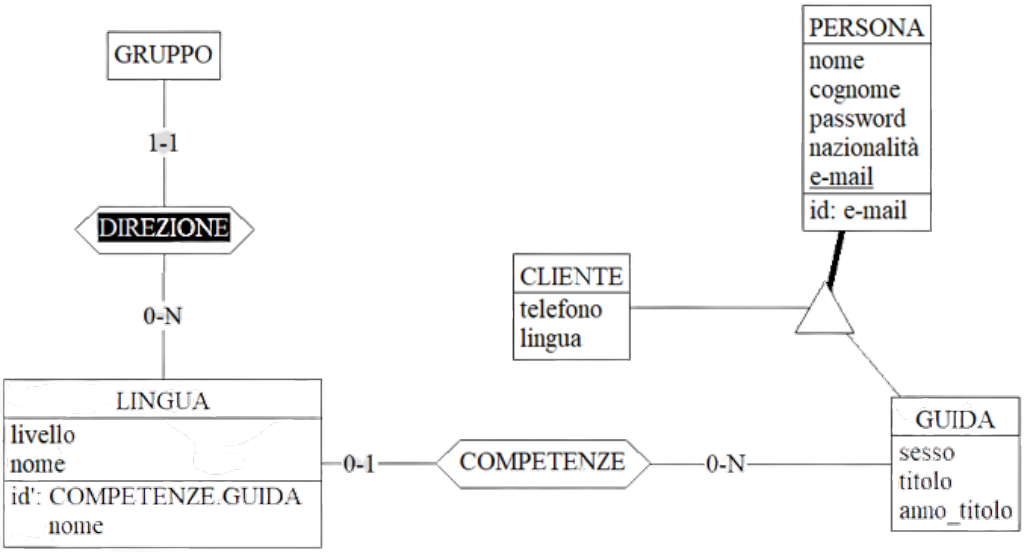


Figura 2.2: l'entità gruppo è descritta nella figura sopra

Per gestire le entità CLIENTE, GUIDA poiché avevano campi simili ho optato per una gerarchia di tipo totale ed Esclusiva (DB_MAIN non mostrava la scritta t, e). Il resto dello schema E-R portato esplicita la modellazione di associare più tipi di lingue ad una determinata guida. Lingua quindi è un'entità debole che necessita della chiave di GUIDA per essere completa. Adottando questo metodo per ogni gurppo posso risalire alla guida e con che lingua e a che livello sarà diretto.

Biglietto-Ordine-Cliente

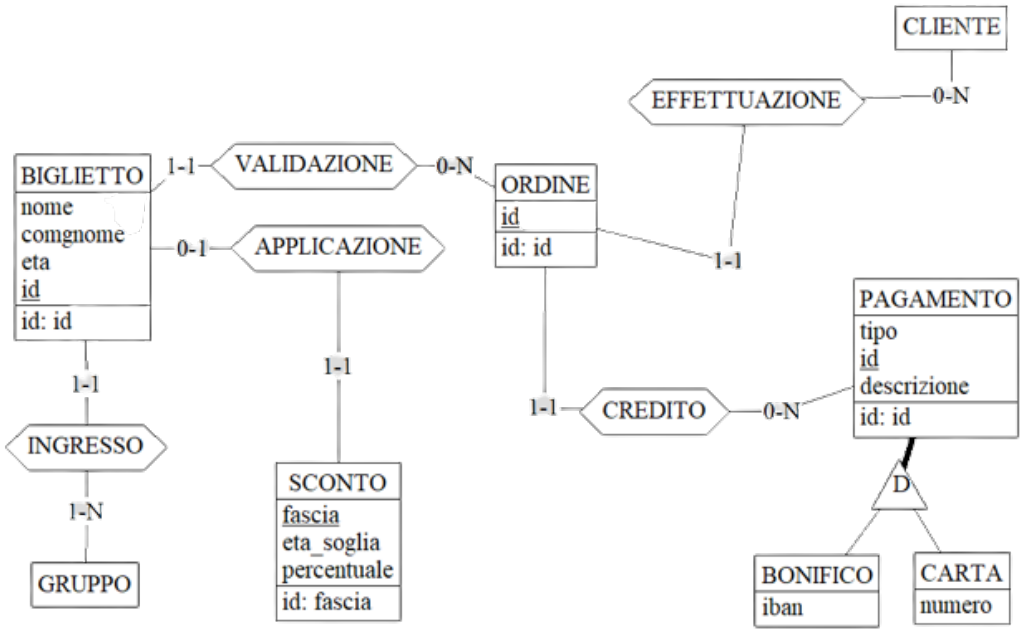
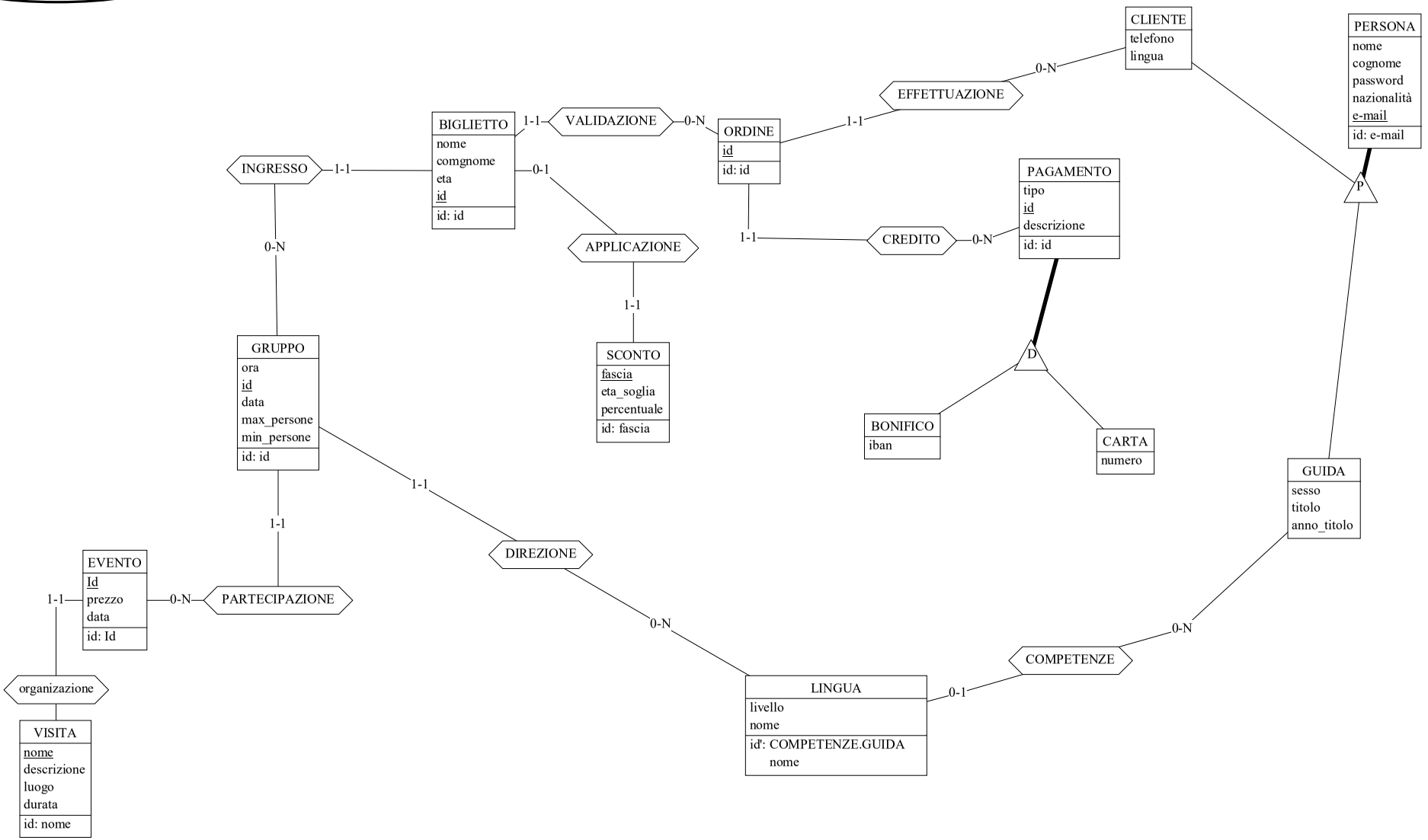


Figura 2.3: i biglietti rappresentano il punto di collegamento relazionale fra i gruppi e i clienti

Per ogni gruppo è previsto che verranno pubblicati N biglietti in base al numero minimo di posti destinabili ad un gruppo. Viste le richieste del testo ho deciso di rendere personale il biglietto e dividere l'acquirente dal proprietario. Lo sconto va verificato per ogni biglietto e il suo acquirente, le fasce di prezzo sono predefinite. Un ordine può riferirsi a più biglietti, in questo modo il cliente potrà acquistare più biglietti. Per il pagamento è stata generata una gerarchia parziale ed esclusiva poiché il pagamento con paypall sarebbe un link al sito e si potrebbe benissimo gestire direttamente lato software.

2.1 Schema scheletro

SCHELETRO_COMPLETO/1



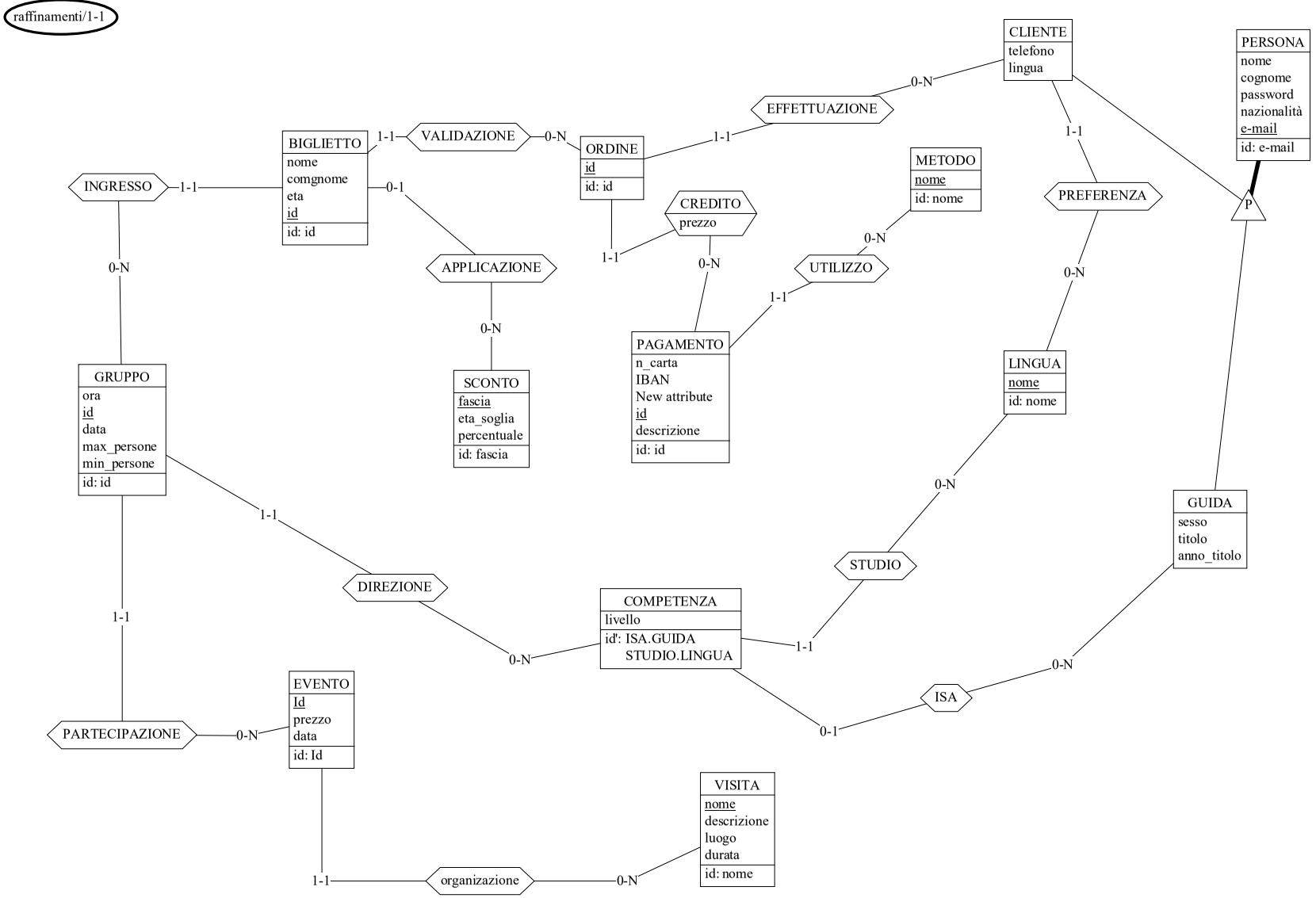
2.2 Raffinamenti proposti

Come si nota dagli schemi riportati, esistono attributi di qualche tabella che sarebbe meglio modellare come entità indipendenti. In Particolare:

- L'attributo 'lingua' dell'entità cliente verrà modellato come entità nuova nello schema contenente tutte le lingue che il dominio del problema dovrà trattare
- l'entità lingua diventerà 'COMPETENZA', sarà un'entità debole poiché avrà come super-chiave il riferimento esterno alla lingua e alla guida.
- l'attributo 'tipo' presente nell' entità 'PAGAMENTO' sarà gestito a parte in modo da definirne i metodi di pagamento supportati. l'entità si chiamerà 'METODO' così facendo posso eliminare la gerarchia nata da pagamento
- la relazione 'CREDITO' ora conterrà un attributo prezzo
- corretta la cardinalità fra sconto e biglietto (0-N) considerando che un tipo di sconto potrebbe essere legato a più biglietti

La trasformazione di questi attributi in entità comporta numerosi vantaggi: diminuisce gli errori in fase d'inserimento e limita il dominio possibile di questi attributi alle sole istanze presenti nelle relative entità.

2.3 Schema concettuale finale



Capitolo 3

Progettazione logica

3.1 Stima del volume dei dati

Nella tabella di seguito è riportato il volume atteso per ciascun costrutto presente nello schema concettuale. Per maggiore chiarezza, rispetto alla classica tabella che si usa per la stima dei volumi è stata aggiunta una colonna per descrivere brevemente alcuni costrutti dal nome ambiguo (specialmente associazioni), in modo da avere immediatamente idea dell'oggetto di cui si sta parlando senza dover trovare il riferimento nello schema concettuale. Inoltre, per garantire maggiore compattezza sono state omesse le stime dei volumi delle associazioni 1-N, in quanto equivalenti ai volumi delle entità che partecipano alle associazioni stesse con cardinalità 1.

Tabella 3.1: Volume dei dati

CONCETTO	COSTRUTTO	VOLUME	DESCRIZIONE
CLIENTE	E	100000	registrazioni di utenti curiosi
BIGLIETTO	E	80000	i biglietti sono 80% degli utenti iscritti

Tabella 3.1: Volume dei dati

CONCETTO	COSTRUTTO	VOLUME	DESCRIZIONE
ORDINE	E	26000	in media un ordine acquista 3 biglietti quindi $\frac{80000}{3}$
GRUPPO	E	16000	considero una media di 5 persone a gruppo
EVENTO	E	3200	ad ogni evento partecipano 5 gruppi
SCONTO	E	13334	la frequenza degli sconti sono pari ad $\frac{1}{6}$
VISITE	E	640	per ogni visita ci sono 5 eventi
PAGAMENTO	E	13000	ogni 2 ordini viene usato lo stesso metodo
METODO	E	10	ad esagerare prevedo 10 metodi diversi per il pagamento

Tabella 3.1: Volume dei dati

CONCETTO	COSTRUTTO	VOLUME	DESCRIZIONE
LINGUA	E	25	operando nella comunità europea considero tutte le lingue parlate all'interno di essa
GUIDA	E	500	ho molti dipendenti
COMPETENZA	E	1500	assumo che una guida conosca almeno 3 lingue

3.2 Descrizione delle operazioni principali e stima della loro frequenza

Di seguito sono riportate le operazioni principali giá individuate in fase di analisi 1.3, alcuni aggiunti in questa fase, che il sistema dovrà supportare, con una stima della loro frequenza di esecuzione.

Operazione	Frequenza
Aggiunta di visite	1/mese
Aggiunta di eventi	5/mese
inserimento di un nuovo cliente	10/giorno
inserimento di una nuova guida	5/anno
aggiunta di una nuova competenza di una guida	2/mese
gestione e riepilogo ordini	50/giorno
Aggiunta di un nuovo gruppo	5/giorno
creazione di biglietti fino al numero minimo di partecipanti ad un gruppo	5/giorno
vendita di biglietti per un determinato cliente	3/giorno
applicazione sconto in base al destinatario del biglietto acquistato	2/giorno
storico degli acquisti	3/mese
controllo del riempimento dei gruppi	10/giorno
assegnazione di una determinata guida ad un gruppo in base alle sue conoscenze	10/giorno
indicazione dei posti rimanenti per le iscrizioni ad un gruppo	40/giorno
aggiunta di una nuova lingua	2/anno
inserimento di un nuovo metodo di pagamento	1/anno
modifica valori Biglietto	1/mese
creazione vista dello storico ordini per ogni cliente	5/mese
Ricerca filtrata di annunci di visite da acquistare in base a diversi parametri scelti	400/giorno
visualizzazione di biglietti ancora invenduti per un determinato evento	500/giorno
eventi disponibili data una determinata lingua	100/giorno
evento che ha venduto di più a fine anno commerciale	1/anno
definire la guida che ha effettuato più visite a fine anno commerciale	1/anno

Di seguito riporto gli schemi delle operazioni, alcune operazioni sono incluse ad altre in modo da rendere più snella la lettura della relazione e le operazioni descritte più corpose

3.3 Schemi di navigazione e tabelle degli accessi

3.3.1 Aggiunta nuovo record cliente e guida, visita

Per quanto riguarda l'aggiunta di guide e clienti, verrà omesso il corrispondente schema di navigazione in quanto coincide con l'entità stessa

CONCETTO	COSTRUTTO	ACCESSI	TIPO
utente	E	1	S
		$TOT = 1S$	

queste valutazioni valgono per ogni tipo di accesso in scrittura della singola entità.

3.3.2 Aggiunta nuovo gruppo

per aggiungere un nuovo gruppo devo eseguire operazioni come:

- lettura dell'evento che si vuole caricare
- lettura della guida che si vuole assegnare
- lettura della lingua che si vuole assegnare
- scrittura di pubblicazione del minimo dei biglietti

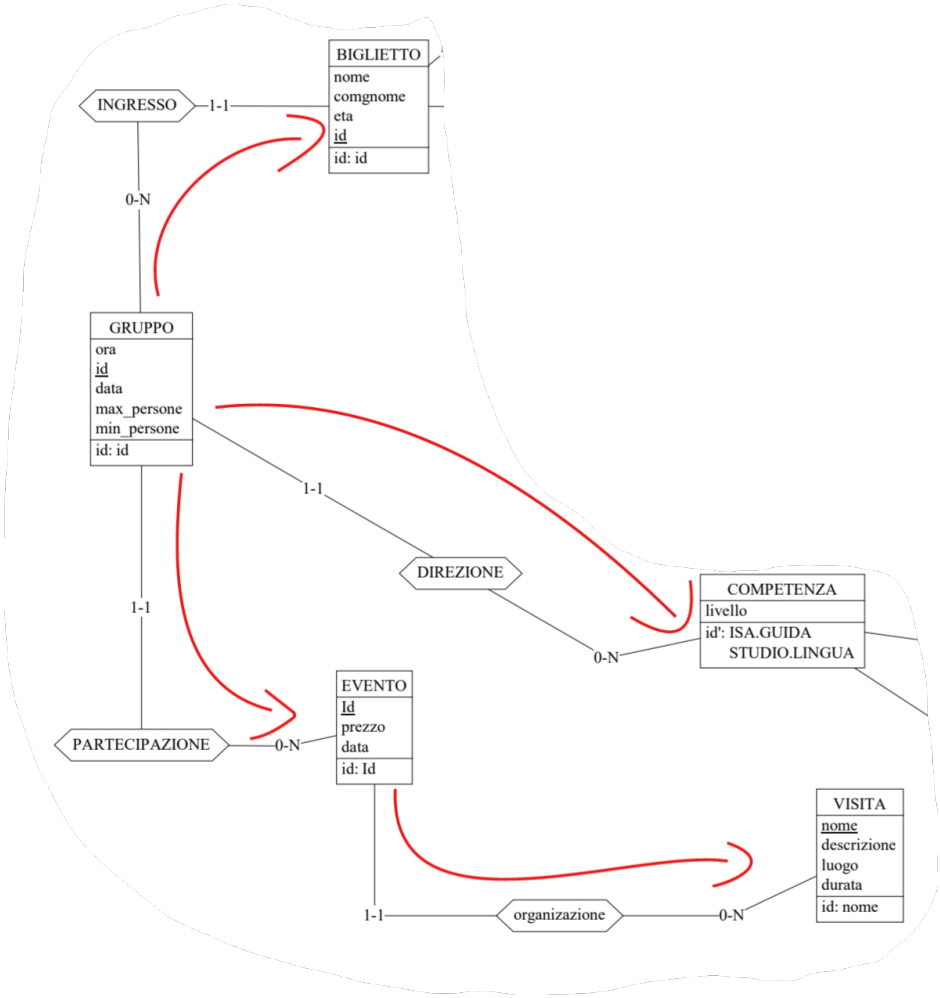


Figura 3.1: la freccia $EVENTO \rightarrow VISITA$ è da non considerarsi poiché evento avrà già un id di VISITA e non sarà necessario leggerlo

CONCETTO	COSTRUTTO	ACCESSI	TIPO
EVENTO	E	1	L
COMPETENZA	E	1	L
BIGLIETTO	E	5	S
		$TOT = 5S + 2L$	

3.3.3 Inserimento delle competenze di una guida

Per questa operazione bisogna ricordarsi che l'entità COMPETENZE è di tipo debole e quindi oltre a fare una scrittura di un nuovo record bisogna

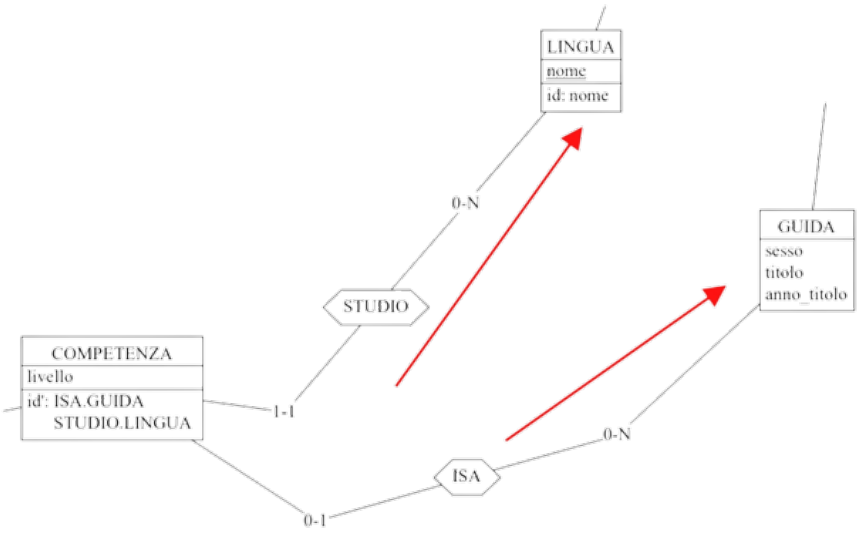


Figura 3.2: Schema di navigazione per la lettura dati chiave competenza

anche fare una lettura per controllare le chiavi delle entità di riferimento.

CONCETTO	COSTRUTTO	ACCESSI	TIPO
COMPETENZE	E	1	S
GUIDA	E	1	L
LINGUA	E	1	L
		$TOT = 1S + 2L$	

3.3.4 Acquisto biglietto per un determinato evento e valutazione sconto

Per l'acquisto di un biglietto bisogna controllare il gruppo a cuoi permette di accedere e visionare l'evento di riferimento, da li è possibile risalire al prezzo del biglietto. dopo di chè sarà possibile eventualmente valutare lo sconto in base a all'età dell'intestatario del biglietto

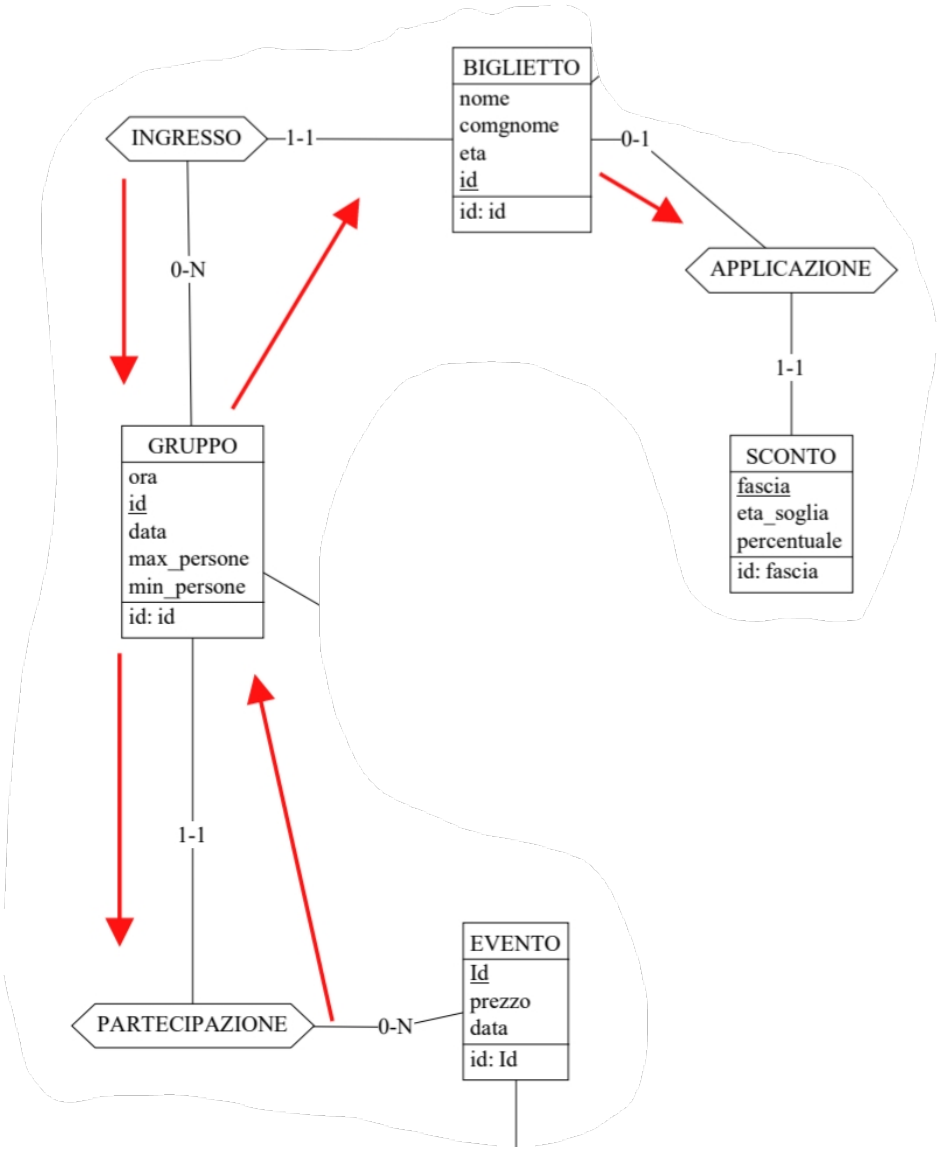


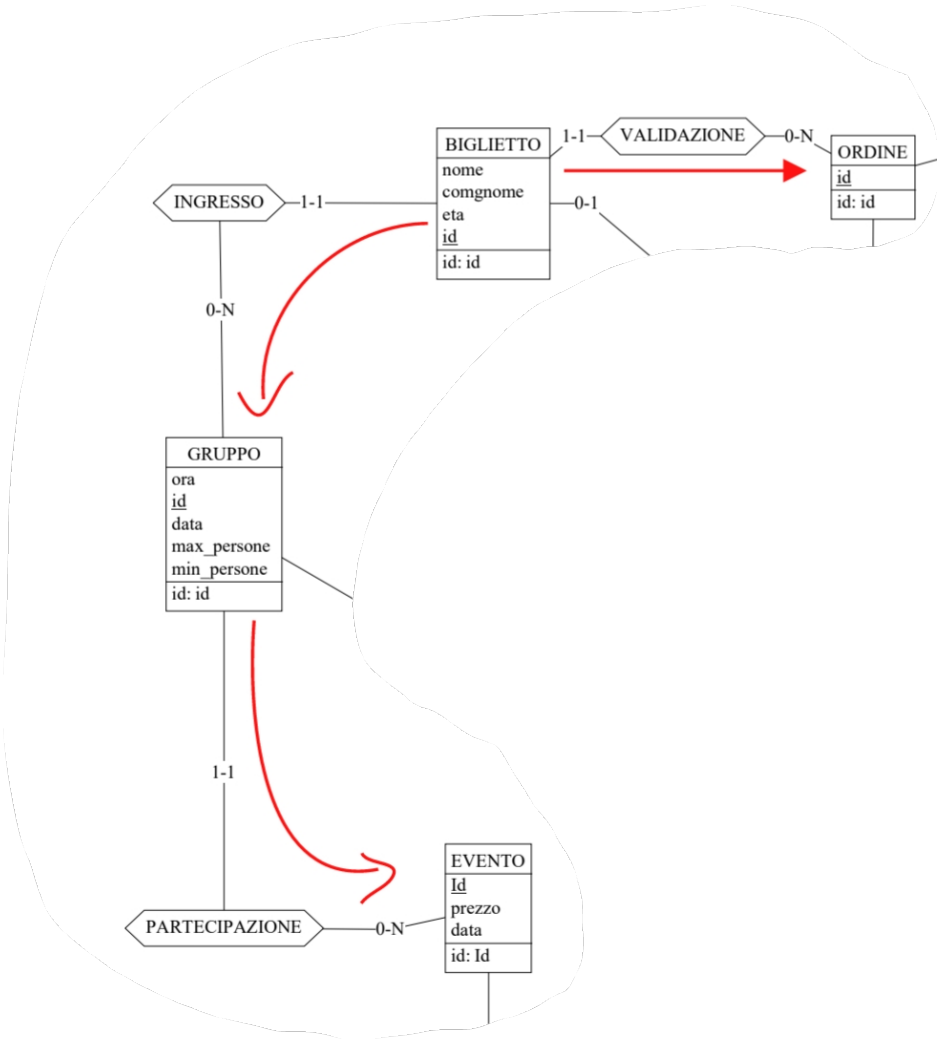
Figura 3.3: Schema di navigazione per l'acquisto di un biglietto

CONCETTO	COSTRUTTO	ACCESSI	TIPO
BIGLIETTO	E	1	S
GRUPPO	E	1	L
EVENTO	E	1	L
SCONTO	E	1	L
		$TOT = 1S + 3L$	

potrebbe sembrare molto macchinoso visto il volume dei dati espresso in precedenza navigare lo schema fino ad evento per definire il prezzo del biglietto, per tanto più avanti verrà indicato l'attributo prezzo in biglietto.

3.3.5 Visualizzazione dei biglietti invenduti per un determinato evento

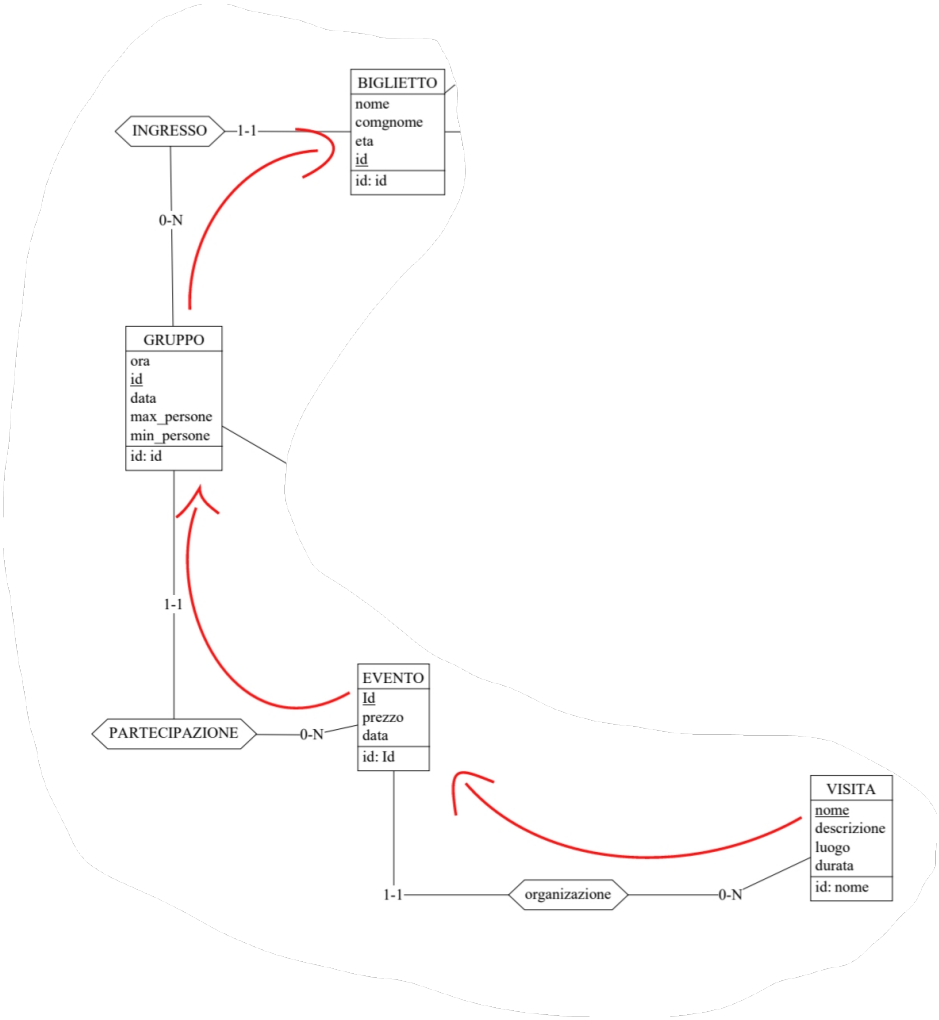
Per la visualizzazione dei biglietti invenduti bisogna prima controllare tutti i biglietti associati ai gruppi dell'evento citato, dopo di che bisogna controllare tutti i biglietti che non sono associati ad un ordine compiuto.



CONCETTO	COSTRUTTO	ACCESSI	TIPO
BIGLIETTO	E	1	L
GRUPPO	E	1	L
EVENTO	E	1	L
ORDINE	E	1	L
		$TOT = 4L$	

3.3.6 Evento che ha venduto di più a fine anno commerciale

Per questa valutazione sarà necessario controllare tutti gli eventi annuali e valutare quanti biglietti sono stati venduti per ogni gruppo e il loro prezzo. Considerando che gli eventi hanno una frequenza di 5/mese in un anno ci sono 60 eventi, quindi per ogni evento bisogna controllare tutti i gruppi associati e controllare quanti biglietti sono stati venduti per i relativi gruppi, considerando quanto riferito nella tabella dei volumi in cui ad ogni evento considero 5 gruppi quindi i gruppi annui saranno 60*5=300 a questi aggiungo un delta maggiorativo che renda sproporzionato il conteggio in modo da avere eventi che hanno ospitato più gruppi quindi 300+40 = 340 che sono i gruppi annui. Faccio lo stesso ragionamento per i biglietti. E 300*5= 1500 sarebbe la media dei biglietti annui per ogni gruppo ma devo crearli non proporzionali quindi aggiungo di 200 biglietti in modo da avere un totale di 1700 biglietti annui.



CONCETTO	COSTRUTTO	ACCESSI	TIPO
BIGLIETTO	E	1700	L
GRUPPO	E	340	L
EVENTO	E	60	L
	TOT=	$1700L + 340L + 60L = 2100$	

Tabella 3.2: Volumi per la valutazione dell'evento che ha venduto di più a fine anno commerciale

alla luce delle operazioni raccolte posso aggiornare le frequenze in base alle operazioni individuate

OPERAZIONE	ACCESSI	FREQUENZA	TOTALE
3.3.1	$1S = 2$	identica alla frequenza indicata per l'operazione	moltiplicare per gli accessi la frequenza indicata
3.3.2	$5S + 2L = 5 * 2 + 2 = 12$	5/giorno	$12 * 5 \rightarrow 60/giorno$
3.3.3	$1s + 2S = 4$	2/mese	$4 * 2 \rightarrow 8/mese$
3.3.4	$1L + 3L \rightarrow 2 + 3 = 5$	3/giorno	$5 * 3 \rightarrow 15/giorno$
3.3.5	$4L$	4/giorno	$4 * 500 \rightarrow 2000/giorno$
3.3.6	$1700L + 340L + 60L = 2100$	1/anno	$2100 * 1 \rightarrow 2100/anno$

3.4 Raffinamento dello schema

In questa sezione mi occuperò del raffinamento dello schema concettuale in vista della realizzazione dello schema logico. In particolare, bisogner'a concentrarsi su una serie di operazioni.

Modifica degli attributi multipli e composti

Lo schema non pensata attributi multipli o composti

Eliminazione delle gerarchie

Lo schema presenta come una gerarchia quella generata da PERSONA che ha come figli CLIENTE e GUIDA. Questa gerarchia può essere eliminata, eseguendo un collasso verso il basso, ottenendo lo schema che segue a pagina successiva. Oltre aver risolto la gerarchia ho provveduto a eliminare il campo presente nella relazione CREDITO inserendolo nell'entità ordine.

Scelta delle chiavi

Per le entità che posseggono più di un identificatore occorre designarne uno come chiave primaria. In particolare.

- Utente poteva essere identificato usando come varie combinazioni di attributi, ad esempio (nome, cognome, data di nascita) ho preferito

usare ma email per poterla sfruttare sia come chiave primaria ma anche come identificativo per l'accesso al sistema informativo

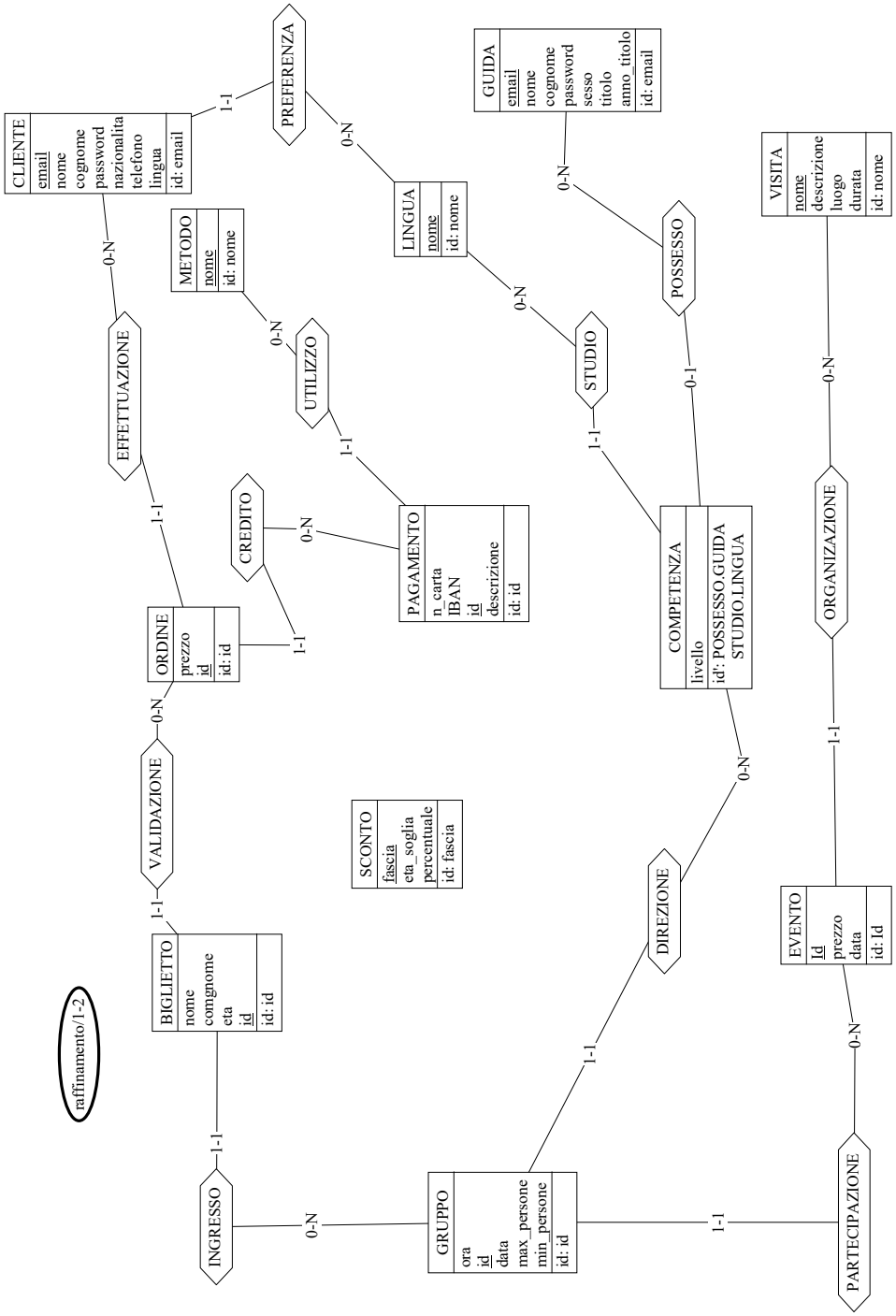
- in Guida è stato fatto lo stesso ragionamento

Eliminazioni degli identificatori esterni

Viene tolto l'attributo "lingua" nell'entità cliente poichè ci sarà il riferimento all'entità lingua attraverso la relazione.
Per lo stesso motivo ho omesso tutti gli altri campi che avrebbero potuto essere identificatori esterni poichè impliciti nella relazione.

Alla pagina seguente è possibile osservare il raffinamento dello schema con le modifiche citate.

Prime modifiche per lo schema logico



3.5 Analisi delle ridondanze

Come già citato a momento della scelta dei volumi vorrei valutare se risulta efficiente la presenza di ridondanze all'interno dello schema.
In particolare nelle operazioni descritte risulta macchinoso dover attraversare più relazioni per ottenere le informazioni necessarie al prezzo del biglietto per assegnare un eventuale sconto o modificare di conseguenza l'importo totale dell'ordine.
Per verificarne i benefici di questa ipotesi, studiamo i casi con e senza ridondanza.

Individuo le operazioni che dipendono dalla scelta di realizzazione di tale attributo.

OPERAZIONE	DESCRIZIONE
3.3.2	aggiunta di un gruppo con creazione 5 biglietti
3.3.4	acquisto di biglietto con valutazione dello sconto

Valutazione operazione 3.3.2

È necessario riferirsi a quella che è la stima del volume dei dati per quanto riguarda la frequenza di questa operazione.
la frequenza di questa operazione è di 60/giorno (dati presi come riferimento alla tabella conclusiva delle frequenze)

Senza ridondanza

i costi dell'operazione sono descritti nel paragrafo apposito

Con ridondanza

con ridondanza bisogna notare che per scrittura di biglietto è necessario accedere al prezzo dell'evento

CONCETTO	COSTRUTTO	ACCESSI	TIPO
EVENTO	E	5	L
COMPETENZA	E	1	L
BIGLIETTO	E	5	S
		$TOT = 5S + 6L$	
ACCESSO		$5 \times 2S + 6L \rightarrow 16$	
FREQUENZA		$16 \times 5 \rightarrow 80/giorno$	

Valutazione operazione 3.3.4

Senza ridondanza

I costi dell'operazione sono descritti nel paragrafo apposito

Con ridondanza

CONCETTO	COSTRUTTO	ACCESSI	TIPO
BIGLIETTO	E	1	S
SCONTO	E	1	L
		$TOT = 1S + 3L$	
ACCESSO		$1 \times 2 + 3L \rightarrow 5$	
FREQUENZA		$5 \times 3 \rightarrow 15/giorno$	

In conclusione considerando che le operazioni senza ridondanza hanno un costo di 75/giorno e quelle con ridondanza 95/giorno, si può concludere che l'aggiunta di ridondanza non sia vantaggiosa

3.6 Traduzione di entità e associazioni in relazioni

Legenda lettura schema logico

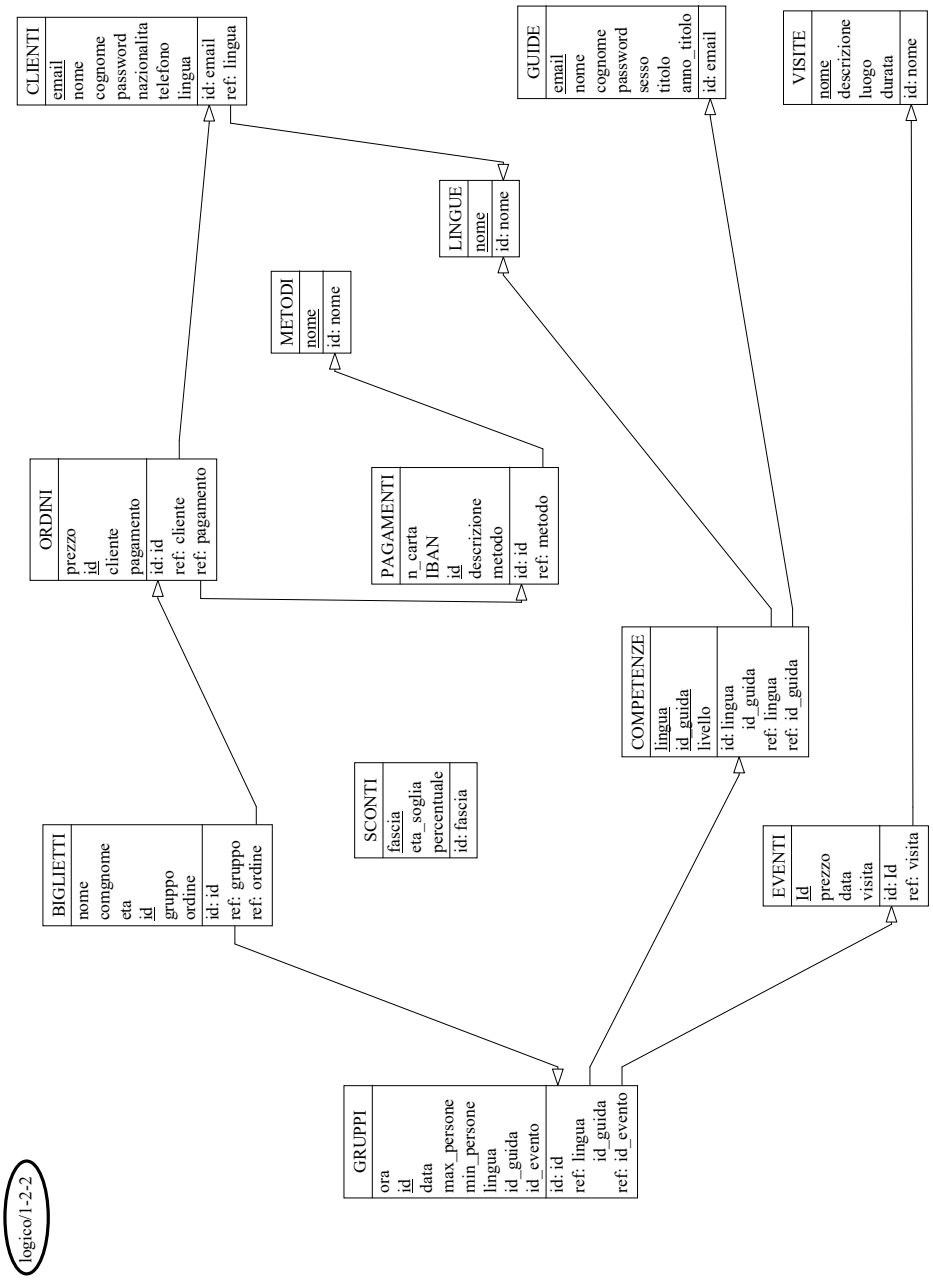
- I campi sottolineati rappresentano una chiava primaria → PK
- I campi con una linea tratteggiata rappresentano una chiave esterna → FK
- i campi che sono sia tratteggiati che sottolineati rappresentano una chiave esterna che funge da chiave primaria → FK+PK

- due campi con una sottolineatura unica rappresentano una chiave primaria composta → PK1+PK2

Dove compaiono chiavi esterne sono riportate le relazioni a cui fanno riferimento. I nomi passeranno dal singolare al plurale per enfatizzare le relazioni tabellari che dovrà rappresentare lo schema

LINGUE(nome)
CLIENTI(email, nome, cognome, password, nazionalita, telefono, lingua*)
FK: lingua REFERENCES lingue
UNIQUE(telefono)
GUIDE(email, nome, cognome, password, sesso, titolo, anno_titolo)
COMPETENZE(guida*, lingua*, livello)
FK: guida REFERENCES guide
FK: lingua REFERENCES lingue
METODI(nome)
PAGAMENTI(id, n_carta, iban, descrizione, metodo*, cliente*)
FK: cliente REFERENCES clienti
FK: metodo REFERENCES metodi
UNIQUE(n_carta, iban)
ORDINE(id, prezzo, pagamento*)
FK: pagamento REFERENCES pagamenti
BIGLIETTI(id, nome, cognome, eta, ordine*, gruppo*)
FK: ordine REFERENCES ordini
FK: gruppo REFERENCES gruppi
GRUPPI(id, ora, data, max_persone, min_persone, lingua*, id_guida*, evento*)
FK: lingua, id_guida REFERENCES competenze
FK: evento REFERENCES eventi
VISITA(nome, descrizione, luogo, durata)
EVENTI(id, prezzo, data, visita*)
FK: visita REFERENCES visite
SCONTO(fascia, eta_soglia, percentuale)

3.7 Schema relazionale finale



3.8 Traduzione delle operazioni in query SQL

Una volta creato il database (DDL in appendiceA) è stato necessario popolarlo con dei dati di esempio per poter testare le query. Ho proceduto nel riempire le tabelle che non avessero alcun tipo di riferimento esterno (quelle generate da entità indipendenti per così dire). Una volta inizializzate ho formulate query per inserire i dati nelle tabelle che avevano riferimenti esterni in modo da poter inserire i dati in modo corretto.

Listing 3.1: popolamento della tabella **COMPETENZE** 3.3.3

```
INSERT INTO COMPETENZE(guida,lingua,livello)
SELECT
    g.email,
    (SELECT l.nome
        FROM LINGUE l
        ORDER BY rand()
        LIMIT 1) as lingua,
    IF(floor(rand()*3)+1 = 1, "B2",
        IF(floor(rand()*3)+1 = 2, "C1", "C2"))
    AS livello
FROM GUIDE g
-- limit 3
;
```

Per creare più varietà possibile nel caso della colonna delle lingue ho usato una subquery che seleziona una lingua a caso dalla tabella LINGUE. Per quanto riguarda il livello ho usato una funzione che genera un numero casuale tra 1 e 3 e in base al valore assegna un livello di competenza.

Listing 3.2: popolamento tabella **EVENTI**

```
INSERT into EVENTI(prezzo,data,visita)
SELECT
    ROUND(15 + RAND() * 15) AS prezzo,
    DATE_FORMAT(DATE_ADD(NOW(), INTERVAL ROUND(RAND()
       *30) DAY), '%Y-%m-%d') as dat,
    (SELECT nome FROM VISITE ORDER by rand() LIMIT 1) as
        visita
FROM DUAL;
```

Questa query inserisce una nuova riga nella tabella EVENTI, usando dei valori generati casualmente.

'SELECT ... from DUAL' è una clausola che seleziona dei valori da una tabella fittizia chiamata DUAL, che serve per eseguire delle operazioni senza usare una tabella vera.
'ROUND(15 + RAND() * 15) AS prezzo' genera un numero casuale fra 15 e 30 e lo arrotonda al valore intero più vicino. Questo valore viene assegnato alla colonna prezzo.
'DATE_FORMAT(DATE_ADD(NOW(), INTERVAL ROUND(RAND()*30) DAY), '%Y-%m-%d') as dat' genera una data casuale tra oggi e 30 giorni dopo, formattata come anno-mese-giorno. Questo valore viene assegnato alla colonna data.
'(SELECT nome FROM VISITE ORDER by rand() LIMIT 1) as visita' seleziona un nome casuale dalla tabella VISITE.

Listing 3.3: popolamento della tabella **GRUPPI**

```
1
2  INSERT INTO GRUPPI(ora,data,max_persone,min_persone,
       lingua,id_guida,id_evento)
3  SELECT
4      -- genera un orario a caso fra le 8 alle 14 e dalle
          16 alle 18 con intervalli di ogni quarto d'ora
5      CASE WHEN
6            RAND() < 0.5 THEN SEC_TO_TIME(FLOOR(RAND() *
                (14 - 8) * 3600 / 900) * 900 + 8 * 3600)
7            ELSE SEC_TO_TIME(FLOOR(RAND() * (18 - 16) *
                3600 / 900) * 900 + 16 * 3600)
8            END AS ora,
9      10 AS max_persone,
10     5 AS min_persone,
11     C.lingua AS lingua,
12     C.guida AS id_guida,
13
14     E.Id AS id_evento
15 FROM
16     -- usa una join tra le tabelle EVENTI e COMPETENZE
          per avere tutte le possibili combinazioni
17     EVENTI E JOIN COMPETENZE C ON TRUE
18 WHERE
19 /*numero a caso fra 0 e il numero totale di righe della
       join*/
20     RAND() * (SELECT COUNT(*) FROM EVENTI E JOIN
          COMPETENZE C ON TRUE) <20;
```

Uso una costante arbitraria per controllare la probabilità d'inserimento: più è alta, più è probabile che venga inserita una riga se è uguale al numero totale di righe della John, viene inserita sempre una riga se è uguale a zero, non viene inserita mai una riga.
Per gestire l'orario in modo che cada ogni quarto d'ora divido l'intervallo di tempo in segmenti di 900 secondi, che corrispondono a 15 minuti. Poi, uso la funzione FLOOR per arrotondare il numero casuale generato da RAND al multiplo inferiore di 900. Infine, uso la funzione SEC_TO_TIME per convertire il numero di secondi in un formato orario.

Listing 3.4: operazione 3.3.2

```
DELIMITER //
    CREATE TRIGGER genera_biglietti AFTER INSERT ON
       GRUPPI
    FOR EACH ROW
    BEGIN
        DECLARE i INT DEFAULT 0;
        WHILE i < 5 DO
            INSERT INTO BIGLIETTI (eta, gruppo) VALUES (
                NEW.id -- id del gruppo appena inserito
            );
            SET i = i + 1;
        END WHILE;
    END //
DELIMITER ;
-- inserisco il nuovo gruppo nella tabella GRUPPI,
   verificando che lingua, id_guida e id_evento siano
   presenti nelle rispettive tabelle
INSERT INTO GRUPPI (ora, max_persone, min_persone,
   lingua, id_guida, id_evento) VALUES (
    '10:00:00',
    20,
    10,
    (SELECT nome FROM LINGUE WHERE nome = 'Italiano'),
       -- subquery per la lingua
    (SELECT email FROM GUIDE WHERE email = 'guida1@email
       .com'), -- subquery per la guida
    (SELECT Id FROM EVENTI WHERE Id = 1) -- subquery per
        l'evento
);
```

Per rispettare le specifiche imposte ho creato un trigger che inserisce 5 biglietti. Un trigger è una procedura speciale che viene eseguita automaticamente

quando si verifica un determinato evento sul database, come l'inserimento, la modifica o la cancellazione di un record. Un trigger può essere usato per implementare dei vincoli di integrità, per registrare delle modifiche, per generare dei valori automatici o per eseguire delle azioni correlate.

Listing 3.5: operazione **Acquisto biglietto per un determinato evento e valutazione sconto** 3.3.4

```
-- Assumiamo che il cliente voglia acquistare un biglietto per l'evento con Id = 1 e che abbia 25 anni
-- Assumiamo anche che il prezzo base dell'evento sia 50 euro

-- Troviamo la fascia di sconto corrispondente all'eta del cliente
SELECT fascia, percentuale FROM SCONTI WHERE eta_soglia <= 25 ORDER BY eta_soglia DESC LIMIT 1;

-- Calcoliamo il prezzo scontato del biglietto
SELECT (SELECT prezzo from EVENTI WHERE id = 200) * (1 - percentuale / 100) AS prezzo_scontato
FROM SCONTI WHERE fascia = (SELECT fascia
                           FROM SCONTI
                           WHERE eta_soglia <= 25
                           ORDER BY eta_soglia DESC
                              LIMIT 1);

-- Inseriamo il biglietto nella tabella BIGLIETTI, associandolo al gruppo con la lingua preferita dal cliente
INSERT INTO BIGLIETTI (eta, gruppo, ordine) VALUES (
    25, -- eta del cliente
    (SELECT id FROM GRUPPI WHERE lingua = (SELECT lingua
        FROM CLIENTI WHERE email = 'cliente@email.com')
       AND id_evento = 1 LIMIT 1), -- id del gruppo con la lingua preferita
    NULL -- ordine non ancora effettuato
);

-- Inseriamo l'ordine nella tabella ORDINI, associandolo al cliente e al pagamento
INSERT INTO ORDINI (prezzo, cliente, pagamento) VALUES (
    (SELECT (SELECT prezzo from EVENTI WHERE id = 200) *
        (1 - percentuale / 100) AS prezzo_scontato
```

```
    FROM SCONTI WHERE fascia = (SELECT fascia
                                FROM SCONTI
                                WHERE eta_soglia <= 25
                                ORDER BY eta_soglia DESC
                                    LIMIT 1)), -- prezzo
                                    scontato del
                                   biglietto
    'edo@dodo.com', -- email del cliente
    (SELECT id FROM PAGAMENTI WHERE metodo = 'Carta di
       credito' AND n_carta = 1234567890123456) -- id
       del pagamento con carta di credito
);

-- Aggiorniamo il biglietto inserendo l'id dell'ordine
   appena effettuato
UPDATE BIGLIETTI SET ordine = (SELECT MAX(id) FROM
   ORDINI) WHERE id = (SELECT MAX(id) );
```

Listing 3.6: operazione **Biglietti Invenduti** 3.3.5

```
-- Assumiamo che l'evento di interesse abbia Id = 1

-- Selezioniamo i biglietti che non hanno un ordine
   associato e che appartengono a un gruppo dell'evento
   con Id = 1
SELECT * FROM BIGLIETTI WHERE ordine IS NULL AND gruppo
   IN (
    SELECT id FROM GRUPPI WHERE id_evento = 1
);
```

Listing 3.7: operazione **Evento che ha venduto di piu' a fini statistici** 3.3.6

```
-- Assumiamo che l'anno commerciale finisca il 31
   dicembre
-- Assumiamo anche che il prezzo dei biglietti sia
   uguale per tutti gli eventi

-- Contiamo il numero di biglietti venduti per ogni
   evento nell'anno commerciale
SELECT  COUNT(*) AS biglietti_venduti
FROM BIGLIETTI,GRUPPI,EVENTI
WHERE ordine IS NOT NULL AND BIGLIETTI.gruppo = GRUPPI.
   id
```

```
    AND GRUPPI.id_evento = EVENTI.Id
    AND EVENTI.data <= '2023-12-31'
    GROUP BY id_evento;

-- Troviamo l'evento con il maggior numero di biglietti venduti
SELECT id_evento, biglietti_venduti FROM (
SELECT id_evento, COUNT(*) AS biglietti_venduti FROM BIGLIETTI WHERE ordine IS
NOT NULL AND data <= '2023-12-31' GROUP BY id_evento
) AS T ORDER BY biglietti_venduti DESC LIMIT 1;
```

Capitolo 4

Progettazione dell'applicazione

L'applicazione è una web app che deve permettere ai visitatori del sito di registrarsi consultare i vari eventi che più possano interessarli e prenotarsi ad uno di questi venendo aggiunti in un gruppo. In questa versione embrionale verranno utilizzati componenti grafici basilari come tra l'altro è richiesto dalla consegna.

Figura 4.1: l'utente puo' visitare la home con tutte le visite disponibili e registrarsi

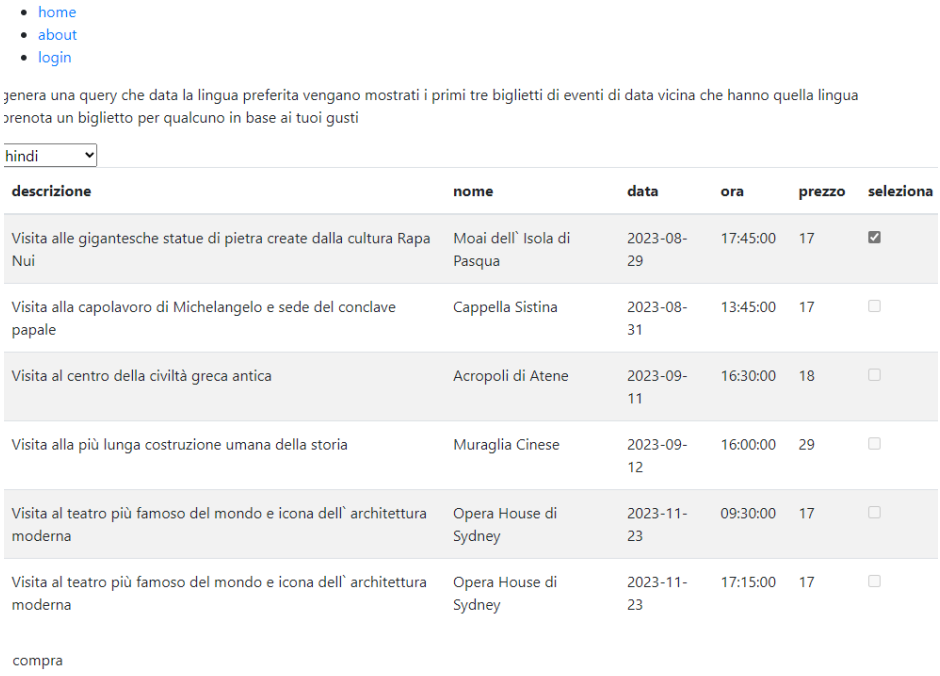


Figura 4.3: selezione all'acquisto di un biglietto

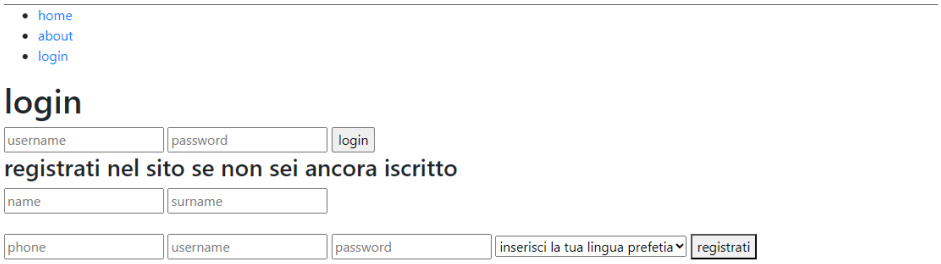


Figura 4.2: l'utente puo' registrarsi o effettuare il login

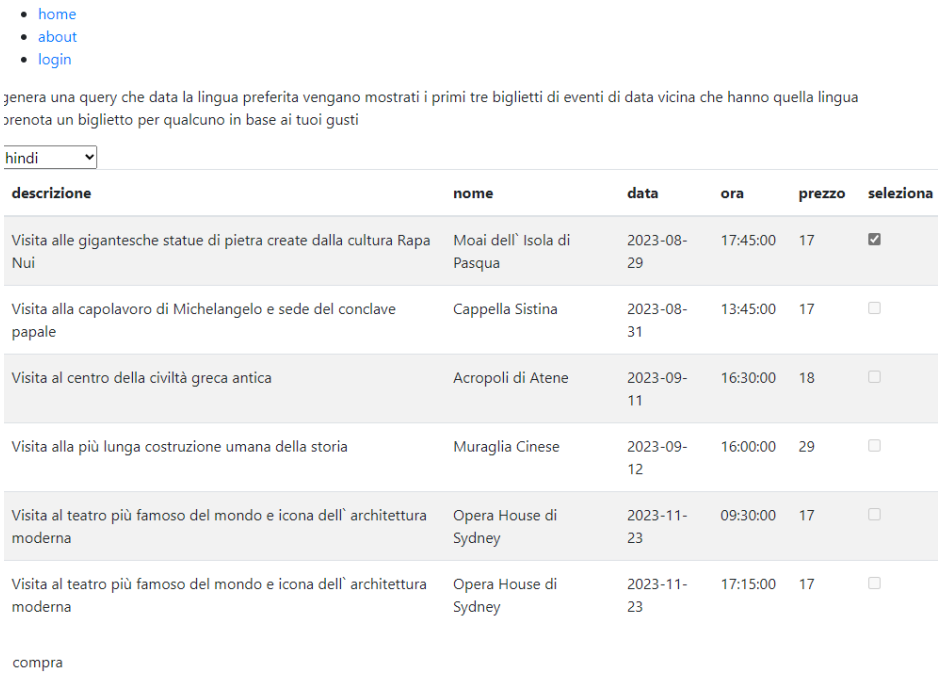


Figura 4.4: dalla dashboard personale l'utente puo' acquistare un biglietto

Appendice A

Codice per la creazione delle tabelle

Listing A.1: si noti dalla linea 90 l'alterazione delle tabelle una volta create per gestire le relazioni definite nel report

```
1  -- crea le query di insert per le tabelle del database
       visite_turistiche
2  create table SCONTI (
3       fascia int AUTO_INCREMENT,
4       eta_soglia int not null,
5       percentuale int not null,
6       constraint IDSCONTO primary key (fascia))ENGINE=
           InnoDB;
7
8  create table LINGUE (
9       nome char(20) not null,
10      constraint IDLINGUA primary key (nome))ENGINE=
           InnoDB;
11
12 create table METODI (
13      nome char(20) not null CHECK (nome in ('Carta di
           credito', 'PayPal', 'Bonifico')),
14      constraint IDMETODO primary key (nome))ENGINE=
           InnoDB;
15
16 create table VISITE (
17      nome char(50) not null,
18      descrizione char(100) not null,
19      luogo char(40) not null,
20      durata time not null,
```

```
21      constraint IDVISITA primary key (nome))ENGINE=
            InnoDB;
22
23  create table CLIENTI (
24      email char(40) not null,
25      nome char(20) not null,
26      cognome char(20) not null,
27      password char(20) not null,
28      nazionalita char(20) not null,
29      telefono int not null UNIQUE,
30      lingua char(20) not null,
31      constraint IDCLIENTE primary key (email))ENGINE=
            InnoDB;
32
33  create table GUIDE (
34      email char(40) not null,
35      nome char(20) not null,
36      cognome char(20) not null,
37      password char(20) not null,
38      sesso char(1) not null CHECK (sesso in ('M', 'F')),
39      titolo char(20) not null CHECK (titolo in ('diploma
            ','Laurea', 'Dottorato', 'Master')),
40      anno_titolo int not null,
41      constraint IDGUIDA primary key (email))ENGINE=
            InnoDB;
42
43  create table COMPETENZE (
44      lingua char(20) not null,
45      guida char(40) not null,
46      livello char(2) not null CHECK (livello in ('B2', '
            C1', 'C2')),
47      constraint IDCOMPETENZA primary key (lingua, guida)
            )ENGINE=InnoDB;
48
49  create table EVENTI (
50      Id int not null AUTO_INCREMENT,
51      prezzo int not null,
52      data DATETIME not null,
53      visita char(20) not null,
54      constraint IDEVENTO primary key (Id))ENGINE=InnoDB;
55
56  create table GRUPPI (
57      ora time not null,
```

```
58      id int not null AUTO_INCREMENT,
59      max_persone int not null,
60      min_persone int not null,
61      lingua char(20) not null,
62      id_guida char(40) not null,
63      id_evento int not null,
64      constraint IDGRUPPO primary key (id))ENGINE=InnoDB;
65
66  create table BIGLIETTI (
67      nome char(20),
68      comgnome char(20),
69      eta int,
70      id int not null AUTO_INCREMENT,
71      gruppo int not null,
72      ordine int,
73      constraint IDBIGLIETTO primary key (id))ENGINE=
            InnoDB;
74
75  create table ORDINI (
76      prezzo int,
77      id int  AUTO_INCREMENT,
78      cliente char(40) not null,
79      pagamento int,
80      constraint IDORDINE primary key (id))ENGINE=InnoDB;
81
82  create table PAGAMENTI (
83      n_carta int UNIQUE,
84      IBAN char(27) UNIQUE,
85      id int not null AUTO_INCREMENT,
86      descrizione  char(100) ,
87      metodo char(20) not null,
88      constraint IDPAGAMENTO primary key (id))ENGINE=
            InnoDB;
89  --alterzaione delle tabelle per i vincoli di integrita'
90  alter table BIGLIETTI add constraint FKINGRESSO
91      foreign key (gruppo) references GRUPPI (id)
92           ON DELETE CASCADE
93           ON UPDATE CASCADE;
94
95  alter table BIGLIETTI add constraint FKVALIDAZIONE
96      foreign key (ordine) references ORDINI (id)
97           ON DELETE CASCADE
98           ON UPDATE CASCADE;
```

```
99
100 alter table CLIENTI add constraint FKPREFERENZA
101      foreign key (lingua) references LINGUE (nome)
102           ON DELETE CASCADE
103           ON UPDATE CASCADE;
104
105 alter table COMPETENZE add constraint FKSTUDIO
106      foreign key (lingua) references LINGUE (nome)
107           ON DELETE CASCADE
108           ON UPDATE CASCADE;
109
110 alter table COMPETENZE add constraint FKPOSSESSO
111      foreign key (guida) references GUIDE (email)
112           ON UPDATE CASCADE
113           ON DELETE CASCADE;
114
115 alter table EVENTI add constraint FKORGANIZAZIONE
116      foreign key (visita) references VISITE (nome)
117           ON DELETE CASCADE
118           ON UPDATE CASCADE;
119
120 alter table GRUPPI add constraint FKDIREZIONE
121      foreign key (lingua, id_guida) references
            COMPETENZE (lingua, guida)
122           ON DELETE CASCADE
123           ON UPDATE CASCADE;
124
125 alter table GRUPPI add constraint FKPARTECIPAZIONE
126      foreign key (id_evento) references EVENTI (Id)
127           ON DELETE CASCADE
128           ON UPDATE CASCADE;
129
130 alter table ORDINI add constraint FKEFFETTUAZIONE
131      foreign key (cliente) references CLIENTI (email)
132           ON DELETE CASCADE
133           ON UPDATE CASCADE;
134
135 alter table ORDINI add constraint FKCREDITO
136      foreign key (pagamento) references PAGAMENTI (id)
137           ON DELETE CASCADE
138           ON UPDATE CASCADE;
139
140 alter table PAGAMENTI add constraint FKUTILIZZO
```

```
141         foreign key (metodo) references METODI (nome)
142             ON DELETE CASCADE
143             ON UPDATE CASCADE;
```

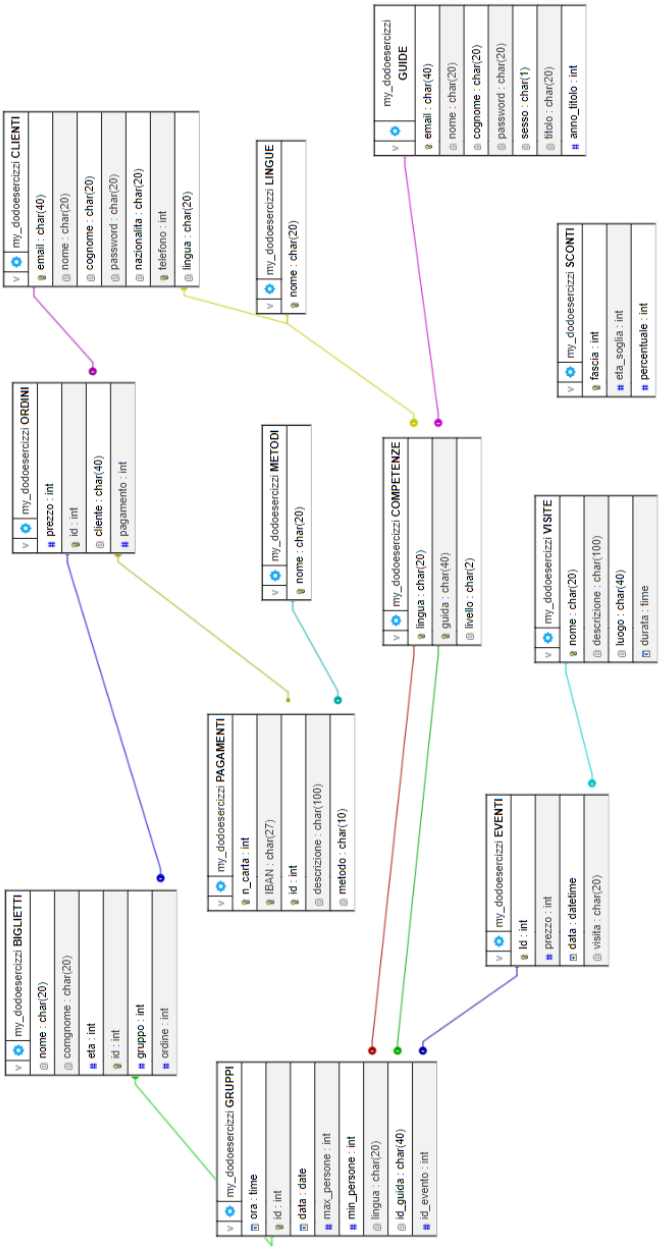


Figura A.1: schema logico generato dall'interfaccia di phpMyAdmin dato il seguente codice SQL