Basi di Dati
Laboratorio

a.a 2024/25

Samuele Caffo 976875
Federico Auda Gioanet 981911

Progettazione Concettuale

1.1 Requisiti Iniziali

Progetto di piattaforma di food delivery

Si deve progettare la base di dati per Cibora (Figura 1(a)), un innovativo servizio di food delivery per gestire i dati dei ristoranti aderenti, degli utenti con i loro relativi ordini e dei fattorini che effettuano le consegne in bicicletta.

Per beneficiare del servizio, ogni utente deve registrarsi inserendo nome, email, password, numero di telefono, indirizzo di recapito. Una volta registratosi, l'utente deve inserire un mezzo di pagamento (es.: carta di credito, paypal, satispay) e ricaricare il proprio borsellino elettronico. Il borsellino ha un saldo che viene aggiornato ad ogni ordinazione e l'utente può ricaricare il proprio borsellino in qualsiasi momento. Inoltre, gli utenti possono sottoscrivere la modalità premium che garantisce una priorità sugli ordini.
L'utente può collezionare codici di sconto da utilizzare al momento dell'ordine in base al numero di ordini effettuati in passato.

Ogni ristorante (Figura 1(b)) è rappresentato da un nome, una descrizione, un indirizzo, il costo della spedizione, un'immagine di profilo e un numero di stellette aggiornato ogni lunedì sulla base della percentuale di recensioni positive dell'ultima settimana. Ogni ristorante appartiene a una o più categorie in base al tipo di cibo offerto (ad esempio: fast food, vegetariano, …).
I ristoranti che dimostrano di saper garantire un ottimo servizio (almeno 20 ordini consegnati correttamente, una valutazione clienti maggiore o uguale a 4.5 stelline su cinque, una percentuale massima di ordini annullati dal ristorante dell'1.5%, una percentuale massima di ordini con reclami del 2.5%) sono considerati Top Partner. I Top Partner compaiono in sezioni dedicate all'interno dell'app mobile Cibora e ricevono uno speciale badge che attesta il loro servizio eccellente, aiutando ad aumentare la credibilità e ottenere la fiducia dei clienti. Per i Top Partner si vuole tenere traccia della data in cui sono entrati a far parte della categoria.
I ristoranti propongono agli utenti una lista di piatti da ordinare. Ogni portata ha un titolo, un'immagine, una lista di ingredienti, una lista di allergeni, il prezzo e un eventuale sconto. Inoltre, ogni piatto appartiene ad una o più liste (es. i più venduti, promozioni, dolci, salato, ecc.).

Ogni utente può selezionare una lista di pietanze ed effettuare l'ordine. Finché non sono affidati ad un rider per la consegna, gli ordini possono essere annullati sia dai clienti, sia dai ristoratori. Nel profilo dell'utente si possono ispezionare gli ordini passati ed eventualmente effettuare dei reclami inviando un messaggio al ristorante.

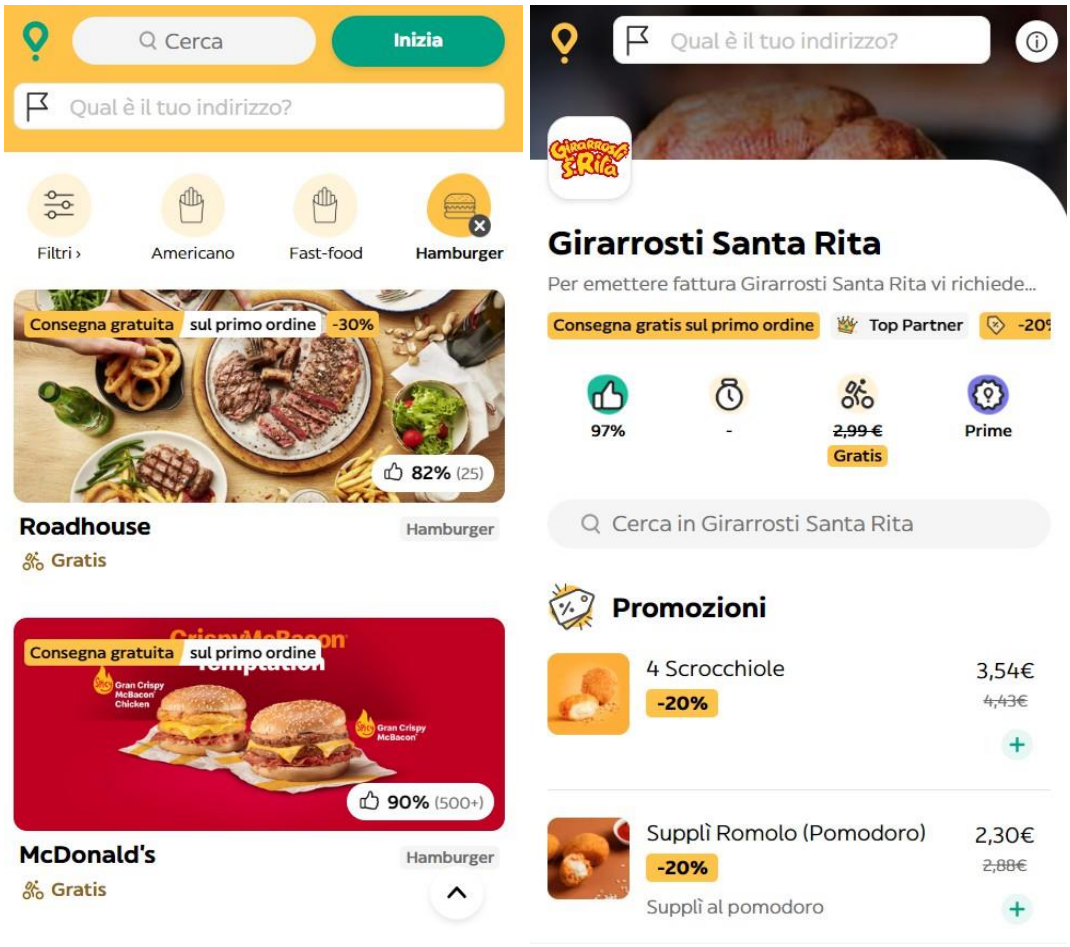


Figura 1 (a) La lista dei ristoranti con filtro "Hamburger". (b) I dettagli di un ristorante.

Il sistema gestisce un numero arbitrario di riders dove ogni rider è identificato da un codice, dallo stato (occupato/disponibile/fuori servizio), dalla posizione aggiornata in tempo reale tramite GPS. I riders sono classificati in base al tipo di mezzo che utilizzano (bicicletta normale, bicicletta elettrica, monopattino). I riders che utilizzano il monopattino devono indicare quanti km possono effettuare prima che si scarichi la batteria.

Al momento dell'ordine, il sistema trova il rider libero con la somma minima della distanza dal ristorante più la distanza dall'utente. Tuttavia, per ordini che prevedano un tragitto "*posizione*

corrente del rider-> ristorante-> cliente" superiore ai 10 km, solo i rider con bici elettrica vengono interpellati. Per monitorare le prestazioni dei ciclofattorini, si vuole tenere traccia del numero di consegne effettuate da ognuno, del momento in cui il cibo da consegnare viene affidato ad un rider e, per le consegne già completate, anche dell'ora in cui l'ordine è stato recapitato al cliente.

Dopo che l'ordine è stato effettuato l'utente ha la possibilità di chattare sia con il ristorante che con il rider in caso ci fossero dei problemi con l'ordine come mancata consegna o netto ritardo.

Quando l'ordine è consegnato l'utente può recensire il ristorante e il rider con una valutazione da 1 a 5 e un commento testuale. Il commento testuale è facoltativo.
Inoltre è anche presente la possibilità di dare una mancia al rider per la consegna.

Una volta al mese, vengono aggiornate le seguenti classifiche:

- Riders più veloci nel consegnare gli ordini
- Cibi più popolari
- Ristoranti con più recensioni positive
- Clienti che hanno speso di più

1.2 Glossario dei Termini

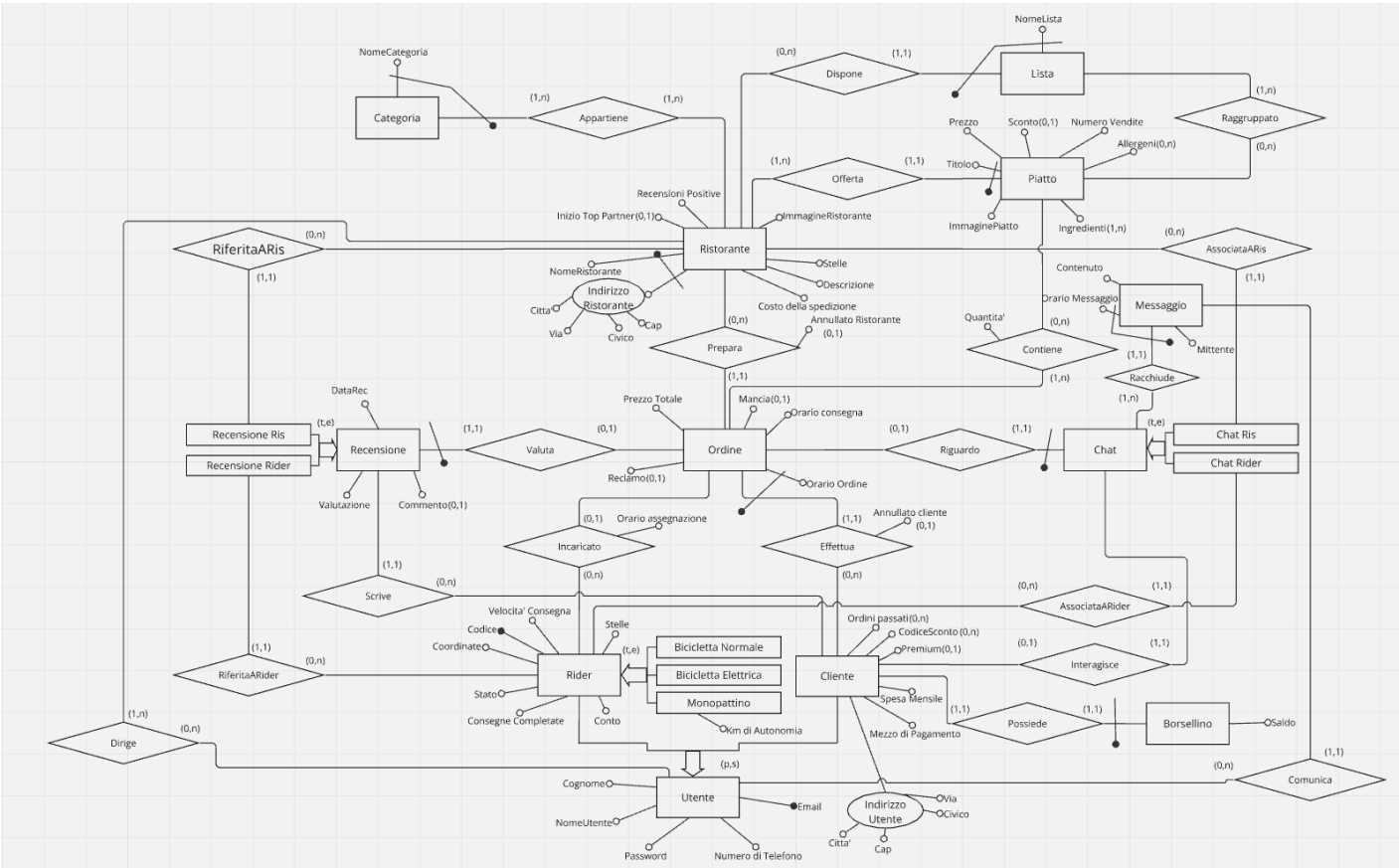
Termine	Descrizione	Sinonimi	Collegamenti
Utente	Persona che si registra sulla piattaforma per effettuare ordini di cibo.	Cliente	Ordine, Borsellino
Ristorante	Struttura commerciale che offre piatti disponibili per l'ordinazione.	-	Ordine
Rider	Persona che effettua le consegne degli ordini di cibo tramite bicicletta o altro	Fattorini, Ciclofattorini	Ordine
Borsellino Elettronico	Conto virtuale dell'utente da cui vengono prelevati i fondi per gli ordini.	Saldo	Utente
Ordine	Acquisto di uno o più piatti effettuato da un utente presso un ristorante.	-	Utente, Ristorante, Consegna
Consegna	Processo di trasporto di un ordine da un ristorante a un cliente, effettuato da un rider.	-	Utente, Ristorante, Ordine
Piatto	Singola portata offerta da un ristorante e ordinabile dagli utenti.	Pietanza, Portata	Ristorante
Chat	Sistema di messaggistica per comunicare con il rider o il ristorante.	-	Utente, Ristorante, Rider
Categoria	Etichetta caratteristica associata a un ristorante		
Top Partner	Ristorante che soddisfa criteri elevati di servizio ed è evidenziato nell'app.	-	Ristorante
Recensione	Valutazione lasciata dall'utente per un ristorante o un rider.	-	Utente, Ristorante, Rider
Lista	Elenco di piatti		

1.3 Requisiti rivisti strutturati in gruppi di frasi omogenee

Frasi di Carattere Generale Si deve progettare la base di dati per un innovativo servizio di food delivery per gestire i dati dei ristoranti aderenti, degli utenti con i loro relativi ordini e dei fattorini che effettuano le consegne.	Frasi relative agli utenti - Ogni utente è rappresentato da nome, e-mail, password, numero di telefono, indirizzo di recapito, mezzo di pagamento e borsellino elettronico. Il borsellino ha un saldo che viene aggiornato ad ogni ordinazione e l'utente può ricaricare il proprio borsellino. - Possono sottoscrivere la modalità premium che garantisce una priorità sugli ordini. - Possono collezionare codici di sconto da utilizzare al momento dell'ordine in base al numero di ordini effettuati in passato. - Possono selezionare una lista di piatti ed effettuare l'ordine. - Possono ispezionare gli ordini passati.
Frasi relative ai ristoranti - Ogni ristorante è rappresentato da nome, descrizione, indirizzo, costo della spedizione, immagine di profilo e numero di stellette aggiornato ogni lunedì sulla base della percentuale di recensioni positive dell'ultima settimana. - Appartiene a una o più categorie in base al tipo di cibo offerto. - Propongono agli utenti una lista di piatti da ordinare. - Quelli che dimostrano di saper garantire un ottimo servizio sono considerati Top Partner, I Top Partner compaiono in sezioni dedicate all'interno dell'app e ricevono uno speciale badge. - Per i Top Partner si vuole tenere traccia della data in cui sono entrati a far parte della categoria. - Ogni ristorante appartiene a una o più categorie in base al tipo di cibo offerto	Frasi relative ai rider - Ogni rider è rappresentato da codice, stato (occupato/disponibile/fuori servizio), posizione aggiornata in tempo reale e tipo di mezzo che utilizzano (bicicletta normale, bicicletta elettrica, monopattino). - Per i riders che utilizzano il monopattino è indicato quanti km possono effettuare prima che si scarichi la batteria. - Si vuole tenere traccia del numero di consegne effettuate da ognuno
Frasi relative agli ordini - Per un ordine il sistema trova il rider libero con la somma minima della distanza dal ristorante più la distanza dall'utente.	Frasi relative ai piatti - Ogni piatto è rappresentato da titolo, immagine, lista di ingredienti, lista di allergeni, prezzo e un eventuale sconto.

- Per ordini che prevedono un tragitto superiore ai 10 km, solo i rider con bici elettrica vengono interpellati. - - Si vuole tenere traccia del momento in cui l'ordine viene affidato ad un rider e dell'ora in cui l'ordine è stato consegnato. - Finché non sono affidati ad un rider per la consegna, gli ordini possono essere annullati sia dai clienti, sia dai ristoratori.	- Ogni piatto appartiene ad una o più liste.
Frasi relative alle recensioni - L'utente può recensire il ristorante e il rider con una valutazione da 1 a 5 e un commento testuale. - Il commento testuale è facoltativo.	Frasi relative alla chat - L'utente ha la possibilità di chattare sia con il ristorante che con il rider

1.4 Schema E-R principale + business rules.



Dizionario dei dati (Entità)

Entita	Descrizione	Attributi	Identificatore
Utente	Persona che si registra sulla piattaforma	NomeUtente, Cognome, Email, Password, Numero di Telefono	Email
Cliente	Persona che effettua ordini di cibo.	NomeUtente, Cognome, Email, Indirizzo Utente, Password, Numero di Telefono, Mezzo di Pagamento, Ordini Passati (0, n), Codice Sconto (0,n), Premim (0,1), Spesa Mensile.	Email
Borsellino	Conto virtuale del cliente da cui vengono prelevati i fondi per gli ordini.	Saldo	Email
Ordine	Acquisto di uno o più piatti effettuato da un utente presso un ristorante.	Prezzo Totale, Orario ordine,Orario Consegna, Reclamo (0,1), Mancia(0,1)	Email, Orario ordine
Rider	Persona che effettua le consegne degli ordini di cibo tramite bicicletta o altro	NomeUtente, Cognome, Email, Password, Numero di Telefono, Stato, Codice, Coordinate, Stelle, Consegne Completate, Conto, Velocità Consegna.	Codice, Email
Bicicletta Normale	Rider che effettua le consegne degli ordini di cibo tramite bicicletta normale	NomeUtente, Cognome, Email, Password, Numero di Telefono, Stato, Codice, Coordinate, Stelle, Consegne Completate, Conto, Velocità Consegna.	Codice, Email
Bicicletta Elettrica	Rider che effettua le consegne degli ordini di cibo tramite bicicletta elettrica	NomeUtente, Cognome, Email, Password, Numero di Telefono, Stelle Stato, Codice, Coordinate,	Codice, Email

		Consegne Completate, Conto, Velocità Consegna.	
Monopattino	Rider che effettua le consegne degli ordini di cibo tramite monopattino	NomeUtente, Cognome, Email, Password, Numero di Telefono, Stato, Codice, Coordinate, Consegne Completate, Conto, Velocità Consegna, Km di Autonomia, Stelle	Codice, Email
Ristorante	Struttura commerciale che offre piatti disponibili per l'ordinazione.	NomeRistorante, Indirizzo Ristorante, InizioTopPartner(0,1), ImmagineRistorante, Stelle, Descrizione, Costo della spedizione, Recensioni positive	NomeRistorante ,Indirizzo Ristorante
Categoria	Indica il tipo di cibo offerto da un ristorante	NomeCategoria	NomeRistorante, Indirizzo Ristorante, NomeCategoria
Piatto	Singola portata offerta da un ristorante e ordinabile dagli utenti.	Titolo, Prezzo, Sconto (0,1), Ingrediente (1,n), Allergeni (0,n), ImmaginePiatto, Numero Vendite.	NomeRistorante, Indirizzo Ristorante, Titolo
Lista	Gruppo di Piatti accomunati da delle caratteristiche	NomeLista	NomeRistorante, Indirizzo Ristorante, Titolo,NomeLista
Recensione	Valutazione lasciata dall'utente per un ristorante o un rider.	Valutazione, Commento (0,1), DataRec	Email, Orario Ordine
Recensione Rider	Valutazione lasciata dall'utente per un rider.	Valutazione, Commento (0,1), DataRec	Email, Orario Ordine
Recensione Ristorante	Valutazione lasciata dall'utente per un ristorante	Valutazione, Commento (0,1), DataRec	Email, Orario Ordine
Chat	Mezzo di comunicazione tra Utente e Rider e tra Utente e Ristorante	-	Email, Orario Ordine

Chat Rider	Mezzo di comunicazione tra Utente e Rider	-	Email, Orario Ordine
Chat Ristorante	Mezzo di comunicazione tra e Utente e Ristorante	-	Email, Orario Ordine
Messaggio	Testo scambiato durato una comunicazione via Chat	Orario Messaggio, Contenuto, Mittente	Email, Orario Ordine, Orario Messaggio

Dizionario dei dati (Associazioni)

Relazioni	Descrizione	Componenti	Attributi
Possiede	Borsellino posseduto dal cliente	Cliente (1,1), Borsellino (1,1)	
Effettua	Cliente effettua gli ordini di cibo	Cliente (0,n), Ordine (1,1)	AnnullatoCliente (0,1)
Scrive	Cliente scrive la recensione per il rider e per il ristorante	Cliente (0, n), Recensione (1,1)	
Racchiude	La Chat racchiude i messaggi contenuti in essa	Chat (1,n), Messaggio (1,1)	
Interagisce	Cliente interagisce con rider e ristorante tramite la chat	Cliente (0,1), Chat (1,1)	
RiferitaARider	Recensione associata a un determinato Rider	Recensione Rider (1,1), Rider (0,n).	
RiferitaARis	Recensione associata a un determinato Ristorante	Recensione Ristorante (1,1), Ristorante (0,n).	
AssociataARider	Chat associata a un determinato Rider	Chat Rider (1,1), Rider (0,n).	

AssociataARis	Chat associata a un determinato Ristorante	Chat Ristorante (1,1), Ristorante (0,n).	
Valuta	La recensione è in riferimento a un determinato ordine	Recensione (1,1), Ordine (0,1)	
Riguardo	La chat è in riferimento a un determinato ordine	Chat (1,1), Ordine (0,1)	
Dirige	Un ristoratore dirige un determinato ristorante	Utente (0,n), Ristorante (1,n)	
Contiene	Un ordine contiene un certo numero di piatti	Ordine (1,n), Piatti(0,n)	Quantità
Prepara	Un ristorante è incaricato della preparazione dei prodotti dell'ordine	Ordine (1,1), Ristorante (0,n)	AnnullatoRistorante (0,1)
Incaricato	Gli ordini vengono incaricati ai Rider per la consegna	Ordine (0,1), Rider (0,n)	OrarioAssegnazione
Comunica	Utente comunica tramite una chat usando i messaggi	Utente (0,n), Messaggio (1,1)	
Offerta	Un ristorante offre alcuni prodotti	Ristorante (1,n) , Piatti (1,1)	
Dispone	Un ristorante può possedere delle liste di prodotti	Ristorante (0,n), Lista (1,1)	
Raggruppato	I prodotti possono essere raggruppati in delle liste	Piatti(0,n), Lista (1,n)	
Appartiene	I ristoranti appartengono a delle categorie	Ristorante (1,n), Categoria (1,n)	

Regole Aziendali

Integritá:

1) Per Ordini che prevedono un tragitto “posizione corrente Rider-> Ristorante -> Cliente” superiore ai 10km, solo i Rider con bici elettrica possono essere interpellati
2) La valutazione nelle recensioni di Ristorante e Rider deve essere un valore compreso tra 1 e 5.
3) In un Ordine ci possono essere solo piatti provenienti dallo stesso ristorante
4) Un ordine può essere annullato o da un cliente o da un ristorante ma non da entrambi.
5) Finché non sono affidati ad un rider per la consegna, gli ordini possono essere annullati sia dai clienti, sia dai ristoratori.

Derivazione:

1) Ogni ristorante ha numero di stellette aggiornato ogni lunedì sulla base della percentuale di recensioni positive dell’ultima settimana.
2) La spesa mensile dell’utente è la somma totale del costo degli ordini effettuati in un mese e viene aggiornata ogni mese
3) La Velocita Consegna di un Rider è la durata media impiegata ad effettuare le sue consegne e si calcola sommando le durate delle consegne del mese diviso il numero di consegne effettuate nel mese
4) Il Numero di vendite di un piatto rappresenta quante volte è stato venduto negli ordini dei Clienti aggiornato ogni mese
5) Le recensioni positive di ogni ristorante si ottiene sommando il numero di recensioni positive ottenute da quel ristorante nel mese passato.
6) Il numero di consegne completate di ogni rider si ottiene sommando il numero di consegne completate da quel rider.
7) I Ristoranti sono considerati Top Partner se hanno almeno 20 ordini consegnati correttamente, una valutazione clienti maggiore o uguale a 4.5 stelline su cinque, una percentuale massima di ordini annullati dal ristorante dell’1.5%, una percentuale massima di ordini con reclami del 2.5%.

Progettazione Logica

2.1 Tavola dei Volumi

Concetto	Tipo	Volume
Utente	E	3.000.000
Cliente	E	2.910.000
Ristoratore	E	75.000
Borsellino	E	2.910.000
Ordine	E	25.000.000
Rider	E	15.000
Bicicletta Normale	E	3.000
Bicicletta Elettrica	E	8.000
Monopattino	E	4.000
Ristorante	E	80.000
Categoria	E	40
Piatto	E	2.000.000
Lista	E	800.000
Recensione	E	8.000.000
Recensione Rider	E	75.000

Recensione Ristorante	E	7.925.000
Chat	E	3.000.000
Chat Rider	E	2.500.000
Chat Ristorante	E	500.000
Messaggio	E	15.000.000
Possiede	A	2.910.000
Effettua	A	20.000.000
Scrive	A	1.000.000
Interagisce	A	2.000.000
RiferitaARider	A	50.000
RiferitaARis	A	7.800.000
AssociataARider	A	2.400.000
AssociataARis	A	100.000
Valuta	A	1.000.000
Riguardo	A	2.000.000
Dirige	A	80.000
Contiene	A	87.500.000
Prepara	A	25.000.000
Incaricato	A	24.900.000
Offerta	A	2.000.000

Dispone	A	780.000
Raggruppato	A	1.800.000
Appartiene	A	240.000
Comunica	A	15.000.000

Motivazioni:
Abbiamo deciso di distribuire i volumi di dati su un arco temporale di 4 anni. Ad esempio, i 25.000.000 di ordini previsti si tradurrebbero in circa 6.250.000 ordini all'anno. Per determinare i volumi "base" del nostro database (come ordini, rider, ristoranti, chat, ecc.), ci siamo ispirati a servizi di food delivery affermati come Glovo e Just Eat. Da questi volumi iniziali, abbiamo poi calcolato altri dati correlati. Ad esempio:

- **Numero totale di piatti nel database**: considerando una media di 25 piatti per ristorante e un totale di 80.000 ristoranti, abbiamo stimato circa 2.000.000 di piatti complessivi nel database.
- **Numero di relazioni "Contiene"**: con una media di 3,5 piatti per ordine e un totale di 25.000.000 di ordini, abbiamo calcolato circa 87.500.000 relazioni "Contiene" tra ordini e piatti.
- **Numero totale di liste**: ipotizzando una media di 10 liste per ristorante e un totale di 80.000 ristoranti, abbiamo stimato circa 800.000 liste nel database.

Questi calcoli ci hanno permesso di avere una visione chiara delle dimensioni del database e di preparare un modello che possa gestire adeguatamente i volumi di dati previsti.

2.2 Tavola delle operazioni

Operazione	Tipo	Frequenza
Iscrizione nuovo utente al sito	I	200/giorno
Cliente effettua un ordine	I	68.000/giorno
Assegnazione di un rider a un ordine	I	67.000/giorno
Aggiornamento della posizione GPS del rider	I	10000/minuto
Gestione dei Pagamenti	I	69.000/giorno
Recensioni e valutazioni	I	4000/giorno
Gestione della Chat tra utenti, ristoratori e rider	I	10000/giorno
Aggiornamento delle stellette dei ristoranti	B	1/settimana
Verifica e aggiornamento dei ristoranti Top Partner	B	1/settimana

MOTIVAZIONI
Abbiamo scelto di analizzare 7 tipi di operazioni interattive e 2 di tipo batch che consideriamo le più costose in termini di risorse e costo operativo.
La frequenza di queste operazioni è stata determinata partendo dai volumi stimati per un periodo di 4 anni. Abbiamo quindi calcolato la frequenza giornaliera dividendo il totale per 4 anni e successivamente per 365 giorni, per ottenere una stima accurata della frequenza giornaliera delle operazioni.

2.3.1 Analisi delle Ridondanze

Elenco di tutte le ridondanze rilevate:

-Attributi ridondanti:

- *Consegne completate* in *Rider*;
- *Velocità consegna* in *Rider*;
- *Prezzo totale* in *Ordine*;
- *Recensioni positive* in *Ristorante*;
- *Ordini passati* in *Cliente*;
- *Numero vendite* in *Piatto*.

-Associazione ridondante:

- *Prepara* tra *Ordine-Ristorante* (Può essere ricavato da *Ordine-Contiene-Piatto*).
- *Dispone* tra *Ristorante-Lista* (Può essere ricavata da *Piatto*-Raggruppato-*Lista*).
- *Interagisce* tra *Cliente-Chat* (Può essere ricavata da *Chat-Riguardo-Ordine-Effettua-Cliente*)
- *Scrive* tra *Cliente-Recensione* (Può essere ricavata da *Recensione-Valuta-Ordine-Effettua-Ordine*)
- *RiferitaARider* tra R*ecensione Rider-Rider* (Può essere ricavata da *Recensione-Valuta-Ordine-Incaricato-Rider*)
- *AssiciataARider* tra *Chat Rider-Rider (*Può essere ricavata da *Chat-Riguardo-Ordine-Incaricato-Rider)*
- *RiferitaARis* tra *Recensione Ristorante-Ristorante* (Si può ricavare da *Recensione-Valuta-Ordine-Prepara-Ristorante*)
- *AssociataARis* tra *Chat Ristorante-Ristorante* (Si può ricavare da *Chat-Riguardo-Ordine-Prepara-Ristorante*)

Nel contesto di un sistema di food delivery, le recensioni sono un elemento cruciale per valutare i servizi e i ristoranti, pertanto è essenziale conservarle anche se l'ordine associato viene eliminato. Al contrario, le chat, pur essendo utili per la risoluzione dei problemi, possono essere considerate meno fondamentali una volta completato il processo di comunicazione e quindi potrebbero essere eliminate senza un impatto significativo. Pertanto, abbiamo deciso di mantenere le associazioni collegate alle recensioni, in quanto sono fondamentali per la qualità del servizio e per la gestione delle valutazioni. Le recensioni continueranno a esistere anche se l'ordine associato viene rimosso, assicurando che i feedback dei clienti rimangano disponibili per future analisi e miglioramenti. D'altro canto, abbiamo scelto di eliminare le associazioni ridondanti collegate alle chat. Questo approccio riduce il costo in termini di spazio e semplifica la gestione del database, mantenendo la coerenza delle informazioni essenziali senza comprometterne l'accessibilità.

Analisi dettagliata della ridondanza *Prezzo Totale*

Tavola dei volumi

Concetto	Tipo	Volume
Ordine	E	25.000.000
Contiene	A	87.500.000

Tavola delle Operazioni

Operazione	Descrizione	Tipo	Frequenza
1	**Memorizza un nuovo ordine**	I	68.000/giorno
2	**Stampa il costo totale dell'ordine**	I	68.000/giorno

Il caso di studio analizza un ordine di 4 piatti (*Contiene* ÷ *Ordini*: 87.500.000÷25000000 = 3.5 ≈ 4 piatti per ordine).

Scenario A: schema di navigazione relativo all'operazione in assenza della ridondanza:

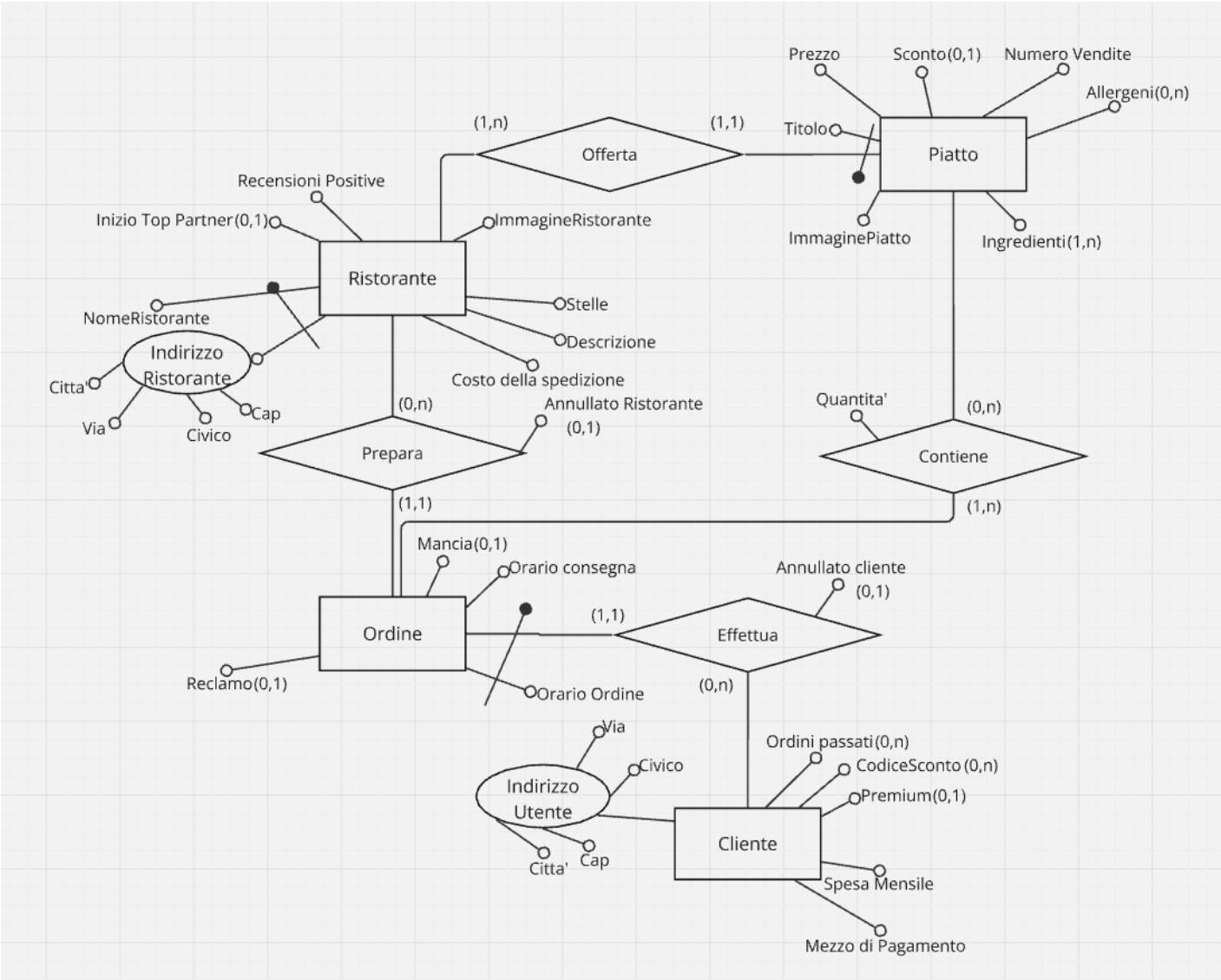


Tavola degli Accessi

Operazione 1

Concetti	Costrutto	Accessi	Tipo
Ordine	**Entità**	1	S
Effettua	**Associazione**	1	S
Prepara	**Associazione**	1	S
Contiene	**Associazione**	4 (o numero di piatti)	S

Operazione 2

Concetti	Costrutto	Accessi	Tipo
Ordine	**Entità**	1	L
Contiene	**Associazione**	4 (o numero di piatti)	L
Piatto	**Entità**	4 (o numero di piatti)	L

Scenario B: schema di navigazione relativo all'operazione in presenza della ridondanza:

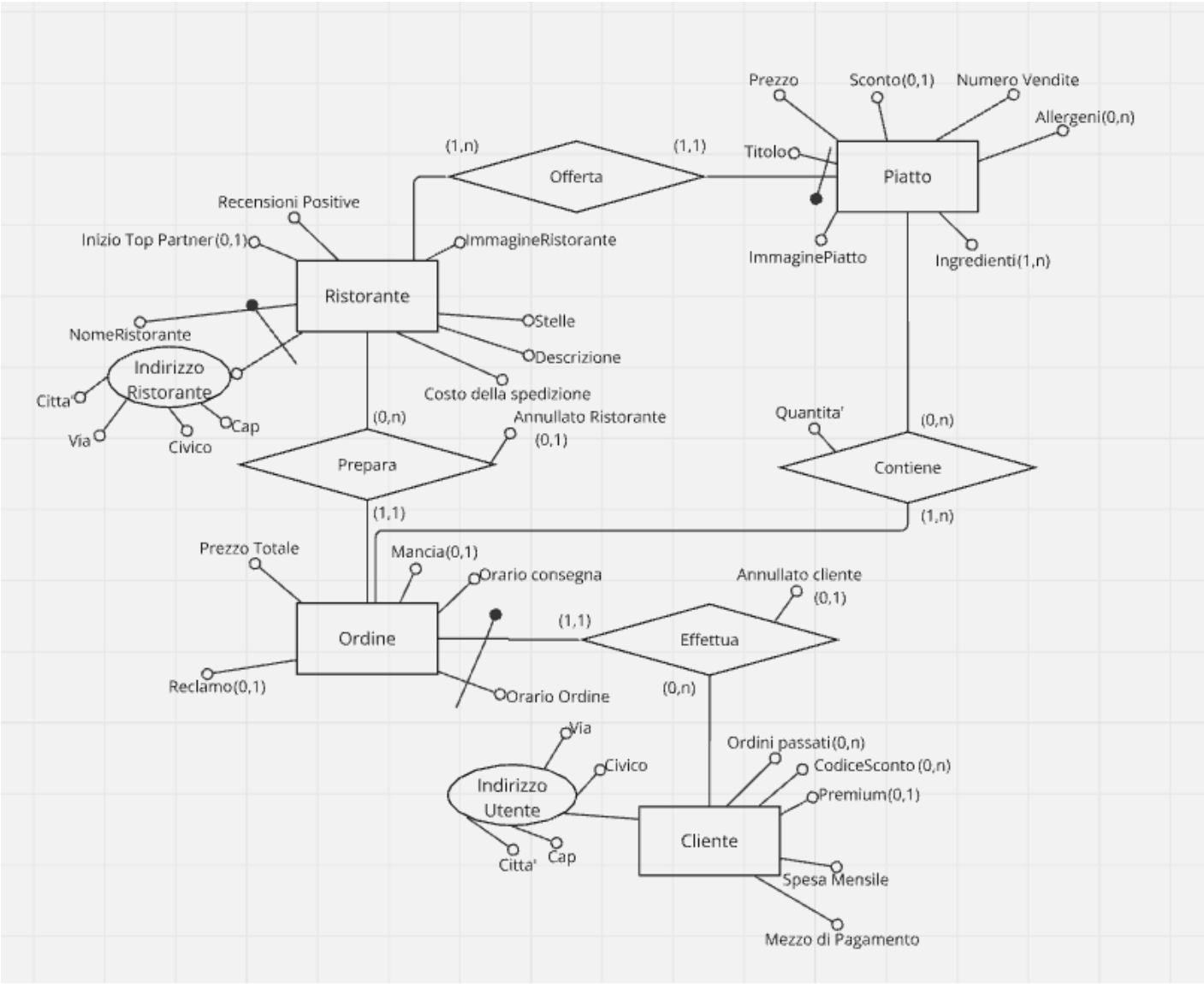


Tavola degli Accessi

Operazione 1

Concetti	Costrutto	Accessi	Tipo
Ordine	**Entità**	1	S
Effettua	**Associazione**	1	S
Prepara	**Associazione**	1	S
Contiene	**Associazione**	4 (o numero di piatti)	S
Piatto	**Entità**	4 (o numeri di piatti)	L
Ordine	**Entità**	4 (o numeri di piatti)	L
Ordine	**Entità**	4 (o numeri di piatti)	S

Operazione 2

Concetti	Costrutto	Accessi	Tipo
Ordine	**Entità**	1	L

Scenario A:
Costi:

- Spazio: 0 Byte
- Tempo:
  - Operazione 1: 7 × 68.000 accessi in scrittura al giorno = 476.000 accessi in scrittura
  - Operazione 2: 9 × 68.000 accessi in lettura al giorno = 612.000 accessi in lettura
  - Contando doppi gli accessi in scrittura: Totale di 1.564.000 accessi

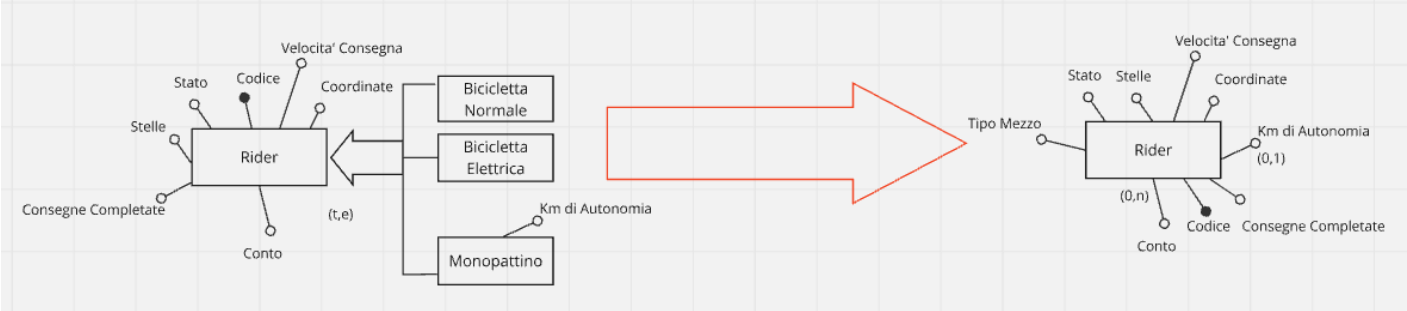
Scenario B:
Costi:

- Spazio: assumendo di usare 4 byte (SMALLFLOAT) per ogni memorizzare il costo totale di ogni ordine: 4 × 68.000 = 272.000 byte = 272 Kb
- Tempo:
  - Operazione 1: ((11 accessi in scrittura × 2) + 8 in lettura) × 68.000 = 2.040.000 accessi
  - Operazione 2: 1 accessi in lettura × 68.000 = 68.000
  - Totale: 2.040.000 + 68.000 = 2.108.000

Togliere la ridondanza ha solamente vantaggi sia a livello di spazio che a livello di access

2.3.2 Eliminazione delle Generalizzazioni

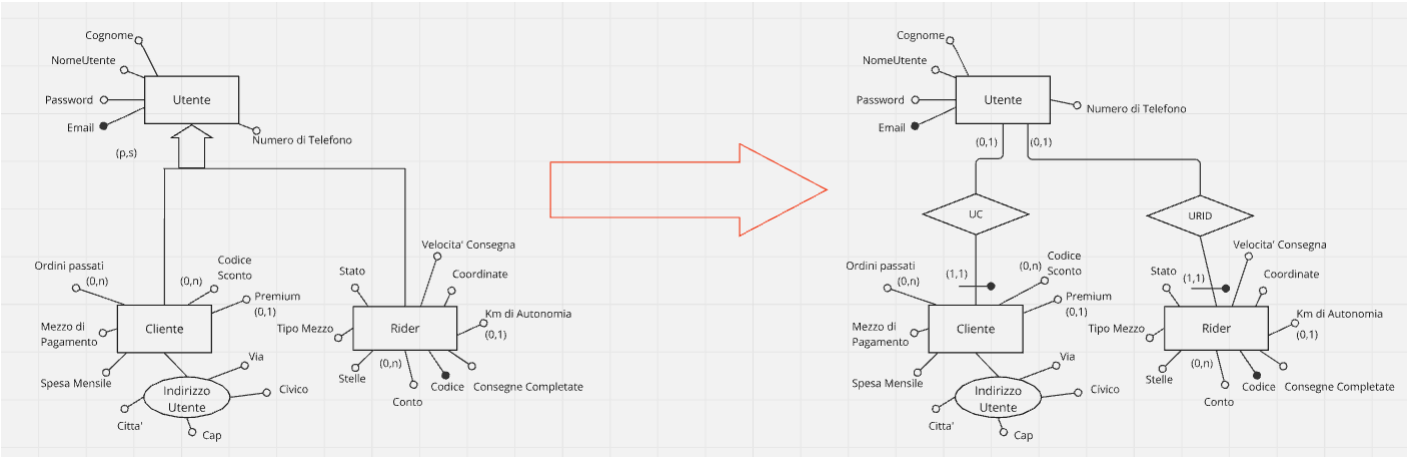
1) Rider



Schema con Generalizzazione Schema senza Generalizzazione

Poiché tutti i rider hanno esattamente un mezzo, è logico mantenere un unico attributo "Tipo di Mezzo", anziché creare relazioni separate.
Questo metodo è flessibile per aggiungere nuovi tipi di mezzi in futuro (es. scooter), senza dover ridefinire la struttura del database.
Le proprietà condivise dai vari tipi di mezzi (come "Velocità Consegna") possono essere centralizzate nell'entità "Rider", evitando duplicazioni.

2) Utente



Schema con Generalizzazione Schema senza Generalizzazione

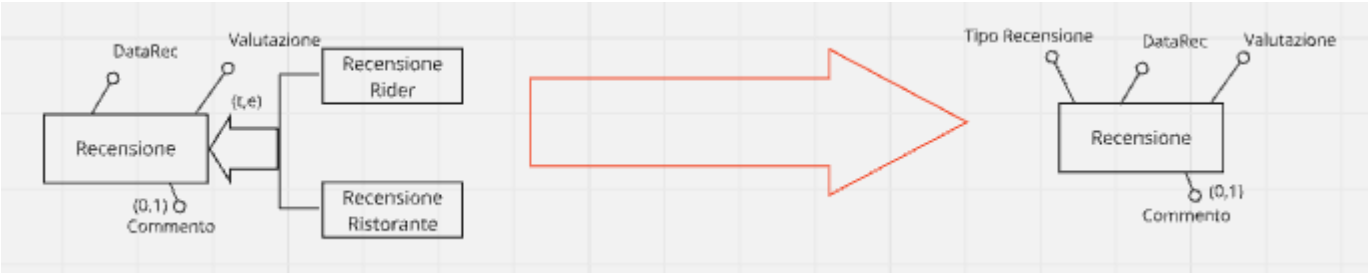
Abbiamo deciso di applicare la Sostituzione della generalizzazione con associazioni poiché molti attributi sono comuni tra "Cliente", "Ristoratore" e "Rider" e separarli nelle entità figlie porterebbe a una duplicazione di dati, in più un utente potrebbe potenzialmente avere più ruoli, come essere sia cliente che ristoratore. Questo tipo di scenario è più gestibile mantenendo l'entità "Utente" centrale e collegando ogni ruolo tramite associazioni.

3) Chat

Schema con Generalizzazione Schema senza Generalizzazione

Abbiamo scelto di unificare "Chat Rider" e "Chat Ristorante" in una singola entità "Chat", ottenendo un modello più semplice e coerente. In questo modo, tutte le chat sono gestite in un'unica entità, riducendo la frammentazione dei dati e semplificando le query. La distinzione tra i tipi di chat viene gestita con un attributo aggiuntivo “TipoChat”.

4) Recensione

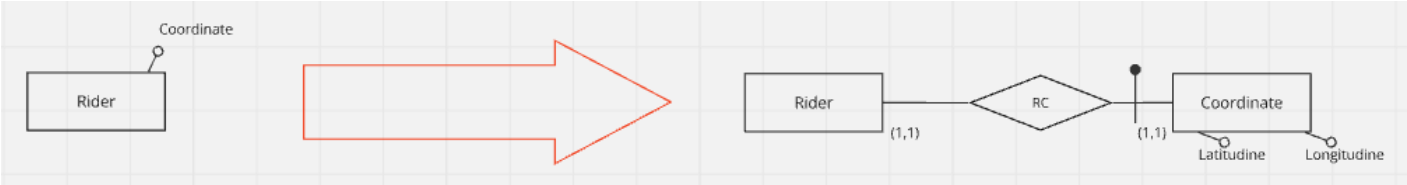


Schema con Generalizzazione Schema senza Generalizzazione

Abbiamo scelto di Accorpare le entità figlie nel genitore perché le recensioni condividono la maggior parte degli attributi, e quindi l’accorpamento semplifica la struttura avendo anche tutte le recensioni gestite in un'unica entità.

2.3.3 Partizionamento/Accorpamento di entità e associazioni

1) Coordinate

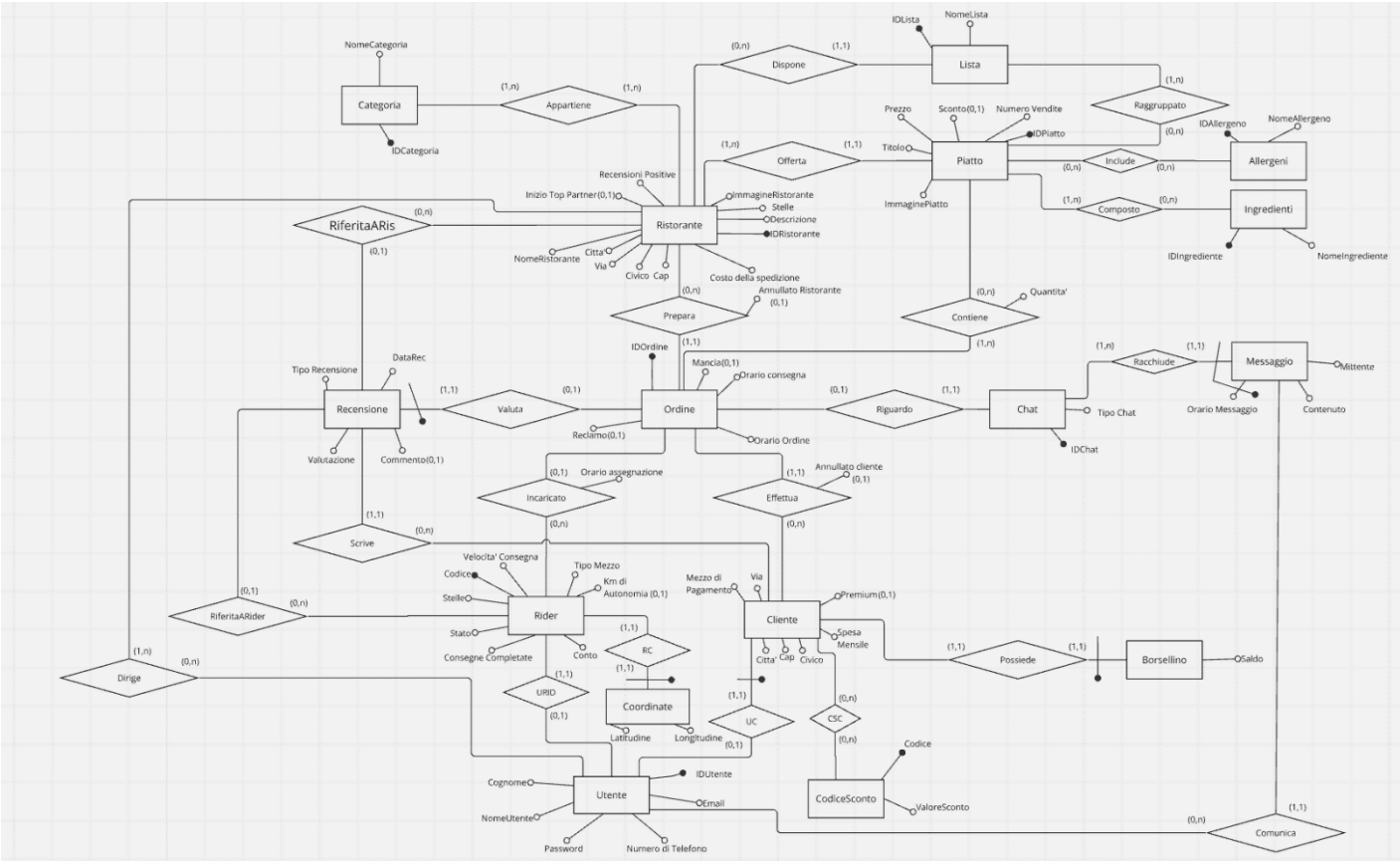


L'aggiornamento delle coordinate del rider è un'operazione ad alta frequenza; quindi, scegliamo di separare gli attributi legati alla posizione in un'entità distinta dato che l'accesso ad altri dati del rider è meno frequente. In questo modo riduciamo il carico sulle query di aggiornamento.

2.3.5 Scelta degli identificatori principali

Abbiamo deciso di adottare gli identificatori "IDRistorante", "IDPiatto", "IDIngrediente", "IDAllergeno", "IDOrdine", "IDUtente", "IDCategoria", "IDLista" e "IDChat" per garantire identificatori semplici e compatti, riducendo la complessità dei campi (come nel caso del Ristorante) e migliorando l'efficienza, soprattutto quando stringhe come i nomi dei piatti o le email risultano troppo lunghe.

2.3.5 Schema E-R Ristruttorato



Regole Aziendali

Integritá:

1) Per Ordini che prevedono un tragitto “posizione corrente Rider-> Ristorante -> Cliente” superiore ai 10km, solo i Rider con bici elettrica possono essere interpellati
2) La valutazione nelle recensioni di Ristorante e Rider deve essere un valore compreso tra 1 e 5.
3) In un Ordine ci possono essere solo piatti provenienti dallo stesso ristorante4)
4) Un ordine può essere annullato o da un cliente o da un ristorante ma non da entrambi.
5) Finché non sono affidati ad un rider per la consegna, gli ordini possono essere annullati sia dai clienti, sia dai ristoratori.
6) TipoMezzo=‘Monopattino’ se e solo se Autonomia ha cardinalità (1,1).

Derivazione:

7) Ogni ristorante ha numero di stellette aggiornato ogni lunedì sulla base della percentuale di recensioni positive dell’ultima settimana.
8) La spesa mensile dell’utente è la somma totale del costo degli ordini effettuati in un mese e viene aggiornata ogni mese
9) La Velocita Consegna di un Rider è la durata media impiegata ad effettuare le sue consegne e si calcola sommando le durate delle consegne del mese diviso il numero di consegne effettuate nel mese
10) Il Numero di vendite di un piatto rappresenta quante volte è stato venduto negli ordini dei Clienti aggiornato ogni mese
11) Le recensioni positive di ogni ristorante si ottiene sommando il numero di recensioni positive ottenute da quel ristorante nel mese passato.
12) Il numero di consegne completate di ogni rider si ottiene sommando il numero di consegne completate da quel rider.
13) I Ristoranti sono considerati Top Partner se hanno almeno 20 ordini consegnati correttamente, una valutazione clienti maggiore o uguale a 4.5 stelline su cinque, una percentuale massima di ordini annullati dal ristorante dell’1.5%, una percentuale massima di ordini con reclami del 2.5%.

Dizionario dei dati (Entità)

Entita	Descrizione	Attributi	Identificatore
Utente	Persona che si registra sulla piattaforma	NomeUtente, Cognome, Email, Password, Numero di Telefono	IDUtente
Cliente	Persona che effettua ordini di cibo.	Mezzo di Pagamento, Premim (0,1), Spesa Mensile, Via, Città, CAP, Civico	IDUtente
Borsellino	Conto virtuale del cliente da cui vengono prelevati i fondi per gli ordini.	Saldo	IDUtente
Ordine	Acquisto di uno o più piatti effettuato da un utente presso un ristorante.	IDOrdine, Orario ordine, Orario Consegna, Reclamo (0,1), Mancia (0,1)	IDOrdine
Rider	Persona che effettua le consegne degli ordini di cibo tramite bicicletta o altro	Stato, Codice, Consegne Completate, Conto, Velocità Consegna, Tipo Mezzo, Stelle, Autonomia (0,1)	Codice
Coordinate	Posizione del Rider	Latitudine, Longitudine	Codice
Ristorante	Struttura commerciale che offre piatti disponibili per l'ordinazione.	IDRistorante, NomeRistorante, Città, Via, CAP, Civico, InizioTopPartner(0,1), ImmagineRistorante, Stelle, Descrizione, Costo della spedizione, Recensioni positive	IDRistorante
Categoria	Indica il tipo di cibo offerto da un ristorante	IDCategoria, NomeCategoria	IDCategoria

Piatto	Singola portata offerta da un ristorante e ordinabile dagli utenti.	IDPiatto, Titolo, Prezzo, Sconto (0,1), ImmaginePiatto, Numero Vendite.	IDPiatto,
Ingrediente	Componente presente in un piatto	IDIngrediente, NomeIngrediente	IDIngrediente
Allergeno	Sostanza che può provocare una reazione allergica nel sistema immunitario.	IDAllergeno, NomeAllergeno	IDAllergeno
Lista	Gruppo di Piatti accomunati da delle caratteristiche	IDLista, NomeLista	IDRistorante
Recensione	Valutazione lasciata dall'utente per un ristorante o un rider.	Tipo Recensione, Valutazione, Commento (0,1), DataRec	IDOrdine
Chat	Mezzo di comunicazione tra Utente e Rider e tra Utente e Ristorante	Tipo Chat, IDChat	IDChat
Messaggio	Testo scambiato durato una comunicazione via Chat	Orario Messaggio, Contenuto, Mittente	IDChat, Orario Messaggio
Codice Sconto	Codice utilizzato dai clienti per ricevere degli sconti sui piatti	Codice, Valore Sconto	Codice

Dizionario dei dati (Associazioni)

Relazioni	Descrizione	Componenti	Attributi
Possiede	Borsellino posseduto dal cliente	Cliente (1,1), Borsellino (1,1)	
Effettua	Cliente effettua gli ordini di cibo	Cliente (0,n), Ordine (1,1)	AnnullatoCliente (0,1)
Scrive	Cliente scrive la recensione per il rider e per il ristorante	Cliente (0, n), Recensione (1,1)	

Racchiude	La Chat racchiude i messaggi contenuti in essa	Chat (1,n), Messaggio (1,1)	
RiferitaARider	Recensione associata a un determinato Rider	Recensione Rider (1,1), Rider (0,n).	
RiferitaARis	Recensione associata a un determinato Ristorante	Recensione Ristorante (1,1), Ristorante (0,n).	
Valuta	La recensione è in riferimento a un determinato ordine	Recensione (1,1), Ordine (0,1)	
Riguardo	La chat è in riferimento a un determinato ordine	Chat (1,1), Ordine (0,1)	
Dirige	Un ristoratore dirige un determinato ristorante	Utente (0,n), Ristorante (1,n)	
Contiene	Un ordine contiene un certo numero di piatti	Ordine (1,n), Piatti(0,n)	Quantità
Prepara	Un ristorante è incaricato della preparazione dei prodotti dell'ordine	Ordine (1,1), Ristorante (0,n)	AnnullatoRistorante (0,1)
Incaricato	Gli ordini vengono incaricati ai Rider per la consegna	Ordine (0,1), Rider (0,n)	OrarioAssegnazione
Comunica	Utente comunica tramite una chat usando i messaggi	Utente (0,n), Messaggio (1,1)	
Offerta	Un ristorante offre alcuni prodotti	Ristorante (1,n) , Piatti (1,1)	
Dispone	Un ristorante può possedere delle liste di prodotti	Ristorante (0,n), Lista (1,1)	
Raggruppato	I prodotti possono essere raggruppati in delle liste	Piatti(0,n), Lista (1,n)	
Appartiene	I ristoranti appartengono a delle categorie	Ristorante (1,n), Categoria (1,n)	
Include	Gli allergeni inclusi in un piatto	Piatto (0,n), Allergeni (0,n)	

Composto	Un piatto composto è composto da ingredienti	Piatto (1,n), Ingrediente (0,n)	
URID	Un Utente che è anche un Rider	Utente (0,1), Rider (1,1)	
UC	Un Utente che è anche un Cliente	Utente (0,1), Cliente (1,1)	
RC	La posizione associata ad ogni Rider	Rider (1,1), Coordinate (1,1)	
CSC	I codici sconto associati ad ogni cliente	Rider (0,n), Coordinate (0,n)	

2.5 Schema relazionale

Ristorante (IDRistorante, Nome, Città, Via, Civico, CAP, Spedizione, Descrizione, Stelle, Immagine, RecensioniPositive, TopPartner*)

Utente (IDUtente, Email, Cognome, Nome, Password, Telefono)

Dirige (Utente, Ristorante)

Rider (Codice, Velocità, Mezzo, Autonomia*, Stato, Consegne, Conto, Utente, Stelle)
Coordinate (Latitudine, Longitudine, Rider)

Cliente (Città, Via, Civico, CAP, MezzoPagamento, Premium*, SpesaMensile, Utente)
CodiceSconto (Codice, Valore)
CSC (Cliente, Codice)
Borsellino (Saldo, Cliente)

Ordine (IDOrdine, Emesso, Completata, Mancia*, Reclamo*, Rider*, Assegnato*, Cliente, AnnullatoCliente*, Ristorante, AnnullatoRistorante*)

Recensione (Tipo, Data, Valutazione, Commento*, Ordine, Cliente, Rider*, Ristorante*)

Chat (Tipo, Ordine, IDChat)
Messaggio (Chat, Orario, Contenuto, Mittente)

Piatto (Titolo, IDRistorante, Prezzo, Immagine, Sconto*, Vendite, IDPiatto,)

Contiene (Ordine, Piatto, Quantità)

Categoria (IDCategoria, NomeCategoria)
Appartiene (Ristorante, Categoria)

Allergeni (IDAllergeno, NomeAllergeno)
Include (Piatto, Allergeno)

Ingredienti (IDIngrediente, NomeIngrediente)
Composto (Piatto, Ingrediente)

Lista (IDLista, Ristorante, Nome)
Raggruppato (Piatto, Lista)

Vincoli di integrità referenziale

Dirige (Utente) referenzia Utente (IDUtente)
Dirige (Ristorante) referenzia Ristorante (IDRistorante)
Rider (Utente) referenzia Utente (IDUtente)
Coordinate (Rider) referenzia Rider (Codice)
(vincolo unique su Coordinate (Rider))
Cliente (Utente) referenzia Utente (IDUtente)
CSC (Cliente) referenzia Cliente (Utente)
CSC (Codice) referenzia CodiceSconto (Codice)
Borsellino (Cliente) referenzia Cliente (Utente)
(vincolo unique su Borsellino (Cliente))
Ordine (Rider) referenzia Rider (Codice)
Ordine (Cliente) referenzia Cliente (Utente)
Ordine (Ristorante) referenzia Ristorante (IDRistorante)
Recensione (Ordine) referenzia Ordine (IDOrdine)
Recensione (Cliente) referenzia Cliente (Utente)
Recensione (Rider) referenzia Rider (Codice)
Recensione (Ristorante) referenzia Ristorante (IDRistorante)
Chat (Ordine) referenzia Ordine (IDOrdine)
Messaggio (Mittente) referenzia Utente (IDUtente)
Messaggio (Chat) referenzia Chat (IDChat)
Piatto (Ristorante) referenzia Ristorante (IDRistorante)
Contiene (Piatto) referenzia Piatto (IDPiatto)
Contiene (Ordine) referenzia Ordine (IDOrdine)
Appartiene (Ristorante) referenzia Ristorante (IDRistorante)
Appartiene (Categoria) referenzia Categoria (IDCategoria)
Include (Piatto) referenzia Piatto (IDPiatto)
Include (Allergeno) referenzia Allergeni (IDAllergeno)
Composto (Piatto) referenzia Piatto (IDPiatto)
Composto (Ingrediente) referenzia Ingredienti (IDIngrediente)
Lista (Ristorante) referenzia Ristorante (IDRistorante)
Raggruppato (Piatto) referenzia Piatto (IDPiatto)
Raggruppato (Lista) referenzia Lista (IDLista)

SQL per la creazione delle Tabelle e dei Tipi

```
CREATE TABLE Ristorante (
    IDRistorante SERIAL PRIMARY KEY,
    Nome VARCHAR(100),
    Città VARCHAR(100),
    Via VARCHAR(100),
    Civico VARCHAR(10),
    CAP VARCHAR(10),
    Spedizione DECIMAL(5, 2),
    Descrizione TEXT,
    Stelle DECIMAL(3,2) DEFAULT 1,
    Immagine BYTEA,
    RecensioniPositive SMALLINT DEFAULT 0,
    TopPartner BOOLEAN DEFAULT FALSE,

    CHECK (CAP ~ '^[0-9][0-9][01589][0-9][0-9]$'),
    CHECK (Spedizione >= 0),
    CHECK (Stelle BETWEEN 1 AND 5)
);

CREATE TABLE Utente (
    IDUtente SERIAL PRIMARY KEY,
    Email VARCHAR(255) NOT NULL UNIQUE,
    Cognome VARCHAR(100) NOT NULL,
    Nome VARCHAR(100) NOT NULL,
    Password VARCHAR(100) NOT NULL,
    Telefono VARCHAR(13),
    CHECK (Telefono ~ '^\d{3} \d{3} \d{4}$'),
    CHECK (Email ~ '^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$')
);

CREATE TABLE Dirige (
    Utente INTEGER REFERENCES Utente(IDUtente),
    Ristorante INTEGER REFERENCES Ristorante(IDRistorante),
    PRIMARY KEY (Utente, Ristorante)
);

CREATE TYPE MezzoDiTrasporto AS ENUM(
    'monopattino',
    'bicicletta',
    'bicicletta elettrica'
);

CREATE TYPE Stato AS ENUM(
    'Occupato',
    'Disponibile',
```

```
    'Fuori servizio'
);

CREATE TABLE Rider (
    Codice SERIAL PRIMARY KEY,
    Velocità REAL DEFAULT 0,
    Mezzo MezzoDiTrasporto,
    Autonomia INTEGER,
    Stato Stato,
    Consegne INTEGER DEFAULT 0,
    Conto DECIMAL(6, 2) DEFAULT 0, --Massimo che può tenere sul conto 9,999.99
    Utente INTEGER REFERENCES Utente(IDUtente),
    Stelle DECIMAL(3,2) DEFAULT 1,
    CHECK (Velocità >= 0),
    CHECK (Autonomia >= 0),
    CHECK (Conto >= 0),
    CHECK (Stelle BETWEEN 1 AND 5)
);

CREATE TABLE Coordinate (
    Latitudine DECIMAL (9, 6),
    Longitudine DECIMAL (9, 6),
    Rider INTEGER REFERENCES Rider(Codice),
    UNIQUE (Rider)
);

CREATE TYPE MetodoDiPagamento AS ENUM(
    'PayPal',
    'Satispay',
    'Contanti',
    'Carta di Credito',
    'Carta di Debito'
);

CREATE TABLE Cliente (
    Città VARCHAR(100),
    Via VARCHAR(100),
    Civico VARCHAR(10),
    CAP VARCHAR(5),
    MezzoPagamento MetodoDiPagamento,
    Premium BOOLEAN DEFAULT FALSE,
    SpesaMensile DECIMAL(7, 2),
    Utente INTEGER REFERENCES Utente(IDUtente) PRIMARY KEY,

    CHECK (CAP ~ '^[0-9][0-9][01589][0-9][0-9]$'),
    CHECK (SpesaMensile >= 0)
);
```

```
CREATE TABLE CodiceSconto (
   Codice VARCHAR(15) PRIMARY KEY ,
   Valore DECIMAL(5, 2),

   CHECK (Valore >= 0)
);

CREATE TABLE CSC (
   Cliente INTEGER REFERENCES Cliente(Utente),
   Codice VARCHAR(15) REFERENCES CodiceSconto(Codice),
   PRIMARY KEY (Cliente, Codice)
);

CREATE TABLE Borsellino (
   Saldo DECIMAL(7, 2) DEFAULT 0,
   Cliente INTEGER REFERENCES Cliente(Utente) PRIMARY KEY,
   UNIQUE (Cliente),

   CHECK (Saldo >= 0)
);

CREATE TYPE TipoRC AS ENUM (
   'Ristorante',
   'Rider'
);

CREATE TABLE Ordine (
   IDOrdine SERIAL PRIMARY KEY,
   Emesso TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   Completata BOOLEAN DEFAULT FALSE,
   Mancia DECIMAL(5, 2) DEFAULT 0,
   Reclamo TEXT DEFAULT NULL,
   Rider INTEGER REFERENCES Rider(Codice),
   Assegnato TIMESTAMP DEFAULT NULL,
   Cliente INTEGER REFERENCES Cliente(Utente),
   AnnullatoCliente BOOLEAN DEFAULT FALSE,
   Ristorante INTEGER REFERENCES Ristorante(IDRistorante),
   AnnullatoRistorante BOOLEAN DEFAULT FALSE,

   CHECK (Mancia >= 0),
   CHECK (NOT(AnnullatoCliente AND AnnullatoRistorante))
);

CREATE TABLE Recensione (
   Tipo TipoRC,
   Data DATE,
```

```
    Valutazione INTEGER,
    Commento TEXT,
    Ordine INTEGER REFERENCES Ordine(IDOrdine) PRIMARY KEY,
    Cliente INTEGER REFERENCES Cliente(Utente),
    Rider INTEGER REFERENCES Rider(Codice) DEFAULT NULL,
    Ristorante INTEGER REFERENCES Ristorante(IDRistorante) DEFAULT NULL,

    CHECK (Valutazione BETWEEN 1 AND 5),
        CHECK (
        (Rider IS NULL AND Ristorante IS NOT NULL AND Tipo = 'Ristorante') OR
        (Rider IS NOT NULL AND Ristorante IS NULL AND Tipo = 'Rider')
    )
);

CREATE TABLE Chat (
    Tipo TIPORC,
    Ordine INTEGER REFERENCES Ordine(IDOrdine),
    IDChat SERIAL PRIMARY KEY
);

CREATE TABLE Messaggio (
    Chat INTEGER REFERENCES Chat(IDChat),
    Orario TIME DEFAULT CURRENT_TIMESTAMP,
    Contenuto TEXT,
    Mittente INTEGER REFERENCES Utente(IDUtente),
    PRIMARY KEY (Chat, Orario)
);

CREATE TABLE Piatto (
    Titolo VARCHAR(100),
    IDRistorante INTEGER REFERENCES Ristorante(IDRistorante),
    Prezzo DECIMAL(5, 2),
    Immagine BYTEA,
    Sconto DECIMAL(5, 2),
    Vendite INTEGER,
    IDPiatto SERIAL PRIMARY KEY,

    CHECK (Prezzo >= 0),
    CHECK (Sconto >= 0),
    CHECK (Vendite >= 0)
);

CREATE TABLE Contiene (
    Ordine INTEGER REFERENCES Ordine(IDOrdine),
    Piatto INTEGER REFERENCES Piatto(IDPiatto),
    Quantità INTEGER DEFAULT 1,
    PRIMARY KEY (Ordine, Piatto)
```

```
);

CREATE TABLE Categoria (
   IDCategoria SERIAL PRIMARY KEY,
   NomeCategoria VARCHAR(100)
);

CREATE TABLE Appartiene (
   Ristorante INTEGER REFERENCES Ristorante(IDRistorante),
   Categoria INTEGER REFERENCES Categoria(IDCategoria),
   PRIMARY KEY (Ristorante, Categoria)
);

CREATE TABLE Allergeni (
   IDAllergeno SERIAL PRIMARY KEY,
   NomeAllergeno VARCHAR(100) NOT NULL
);

CREATE TABLE Include (
   Piatto INTEGER REFERENCES Piatto(IDPiatto),
   Allergeno INTEGER REFERENCES Allergeni(IDAllergeno),
   PRIMARY KEY (Piatto, Allergeno)
);

CREATE TABLE Ingredienti (
   IDIngrediente SERIAL PRIMARY KEY,
   NomeIngrediente VARCHAR(100) NOT NULL
);

CREATE TABLE Composto (
   Piatto INTEGER REFERENCES Piatto(IDPiatto),
   Ingrediente INTEGER REFERENCES Ingredienti(IDIngrediente),
   PRIMARY KEY (Piatto, Ingrediente)
);

CREATE TABLE Lista (
   IDLista SERIAL PRIMARY KEY,
   Ristorante INTEGER REFERENCES Ristorante(IDRistorante),
   Nome VARCHAR(100) NOT NULL
);

CREATE TABLE Raggruppato(
   Piatto INTEGER REFERENCES Piatto(IDPiatto),
   Lista INTEGER REFERENCES Lista(IDLista),
   PRIMARY KEY (Piatto,Lista)
);
```

Creazione dei Trigger

```
-- Crea la funzione per inserire coordinate con valori NULL quando viene inserito un nuovo Rider
CREATE OR REPLACE FUNCTION insert_coordinate_on_rider_insert()
RETURNS TRIGGER AS $$
BEGIN
   INSERT INTO Coordinate (Latitudine, Longitudine, Rider)
   VALUES (NULL, NULL, NEW.Codice);

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

-- Crea il trigger associato alla funzione quando viene inserito un nuovo Rider
CREATE TRIGGER trigger_insert_coordinate
AFTER INSERT ON Rider
FOR EACH ROW
EXECUTE FUNCTION insert_coordinate_on_rider_insert();

--Verifica che il piatto associato all'ordine appartenga al ristorante associato all'ordine
CREATE OR REPLACE FUNCTION check_plate_restaurant()
RETURNS TRIGGER AS $$
DECLARE
   ristoranteOrdine INT;
   ristorantePiatto INT;
BEGIN
   -- Ottieni il ristorante associato all'ordine
   SELECT Ristorante
   INTO ristoranteOrdine
   FROM Ordine
   WHERE IDOrdine = NEW.Ordine;

   -- Controlla se l'ordine esiste
   IF NOT FOUND THEN
      RAISE EXCEPTION 'Ordine % non trovato.', NEW.Ordine;
   END IF;

   -- Ottieni il ristorante associato al piatto
   SELECT IDRistorante
   INTO ristorantePiatto
   FROM Piatto
   WHERE IDPiatto = NEW.Piatto;

   -- Controlla se il piatto esiste
```

```
   IF NOT FOUND THEN
      RAISE EXCEPTION 'Piatto % non trovato.', NEW.Piatto;
   END IF;

   -- Verifica che il ristorante del piatto sia lo stesso dell'ordine
   IF ristoranteOrdine IS DISTINCT FROM ristorantePiatto THEN
       RAISE EXCEPTION 'Il piatto % non appartiene al ristorante dell''ordine %. Ristorante del piatto: %,
ristorante dell''ordine: %',
                   NEW.Piatto, NEW.Ordine, ristorantePiatto, ristoranteOrdine;
   END IF;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER check_plate_restaurant_trigger
BEFORE INSERT ON Contiene
FOR EACH ROW
EXECUTE FUNCTION check_plate_restaurant();

--Per verificare che la lista e il piatto raggruppati appartengano allo stesso ristorante
CREATE OR REPLACE FUNCTION check_plate_restaurant_in_group()
RETURNS TRIGGER AS $$
DECLARE
   ristorantePiatto INT;
   ristoranteLista INT;
BEGIN
   -- Ottieni il ristorante associato all'ordine
   SELECT IDRistorante
   INTO ristorantePiatto
   FROM Piatto
   WHERE IDPiatto = New.Piatto;

   SELECT Ristorante
   INTO ristoranteLista
   FROM Lista
   WHERE IDLista = New.Lista;

   -- Verifica che il ristorante del piatto sia lo stesso dell'ordine
   IF NOT (ristorantePiatto = ristoranteLista  ) THEN
      RAISE EXCEPTION 'Hai provato a inserire il piatto % (appartenente al ristorante %) alla lista % che però
appartiene al ristorante %.',
                   NEW.Piatto,ristorantePiatto, NEW.Lista, ristoranteLista;
   END IF;

   RETURN NEW;
END;
```

```
$$ LANGUAGE plpgsql;

CREATE TRIGGER check_plate_restaurant_in_group_trigger
BEFORE INSERT ON Raggruppato
FOR EACH ROW
EXECUTE FUNCTION check_plate_restaurant_in_group();

--Per Verificare che la recensione sia effettivamente del rider o del ristorante dell'ordine
CREATE OR REPLACE FUNCTION verifica_RR()
RETURNS TRIGGER AS $$
DECLARE
   IDR INT;
BEGIN
   IF(NEW.Tipo = 'Ristorante') THEN
      --Recupero il ristorante dell'ordine
      SELECT Ristorante
      INTO IDR
      FROM Ordine
      WHERE IDOrdine = NEW.Ordine;

   END IF;
   IF(NEW.Tipo = 'Rider') THEN
      --Recupero il rider dell'ordine
      SELECT Rider
      INTO IDR
      FROM Ordine
      WHERE IDOrdine = NEW.Ordine;
   END IF;

   --Confronto con il rider/Ristorante della recensione
   IF NOT(IDR = NEW.Rider) OR NOT(IDR = NEW.Ristorante) THEN
      RAISE EXCEPTION 'Il % % che hai inserito nella recensione non è quello associato all ordine, intendevi
forse % ',
         NEW.Tipo,NEW.Rider,IDR;
   END IF;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger
AFTER INSERT ON Recensione
FOR EACH ROW
EXECUTE FUNCTION verifica_RR();

--Per aggiornare il punteggio delle recensioni dei ristoranti
CREATE OR REPLACE FUNCTION aggiorna_media_recensioni()
```

```
RETURNS TRIGGER AS $$
BEGIN
   IF (NEW.Tipo = 'Ristorante') THEN
      -- Calcolo della nuova media delle recensioni per il ristorante specificato
      UPDATE Ristorante
      SET Stelle = (
         SELECT ROUND(AVG(Valutazione), 2)
         FROM Recensione
         WHERE Ristorante = NEW.Ristorante
      ),
      RecensioniPositive = RecensioniPositive + CASE WHEN NEW.Valutazione >= 3 THEN 1 ELSE 0 END
      WHERE IDRistorante = NEW.Ristorante;
   END IF;

   IF (NEW.Tipo = 'Rider') THEN
      -- Calcolo della nuova media delle recensioni per il rider specificato
      UPDATE Rider
      SET Stelle = (
         SELECT ROUND(AVG(Valutazione), 2)
         FROM Recensione
         WHERE Rider = NEW.Rider
      )
      WHERE Codice = NEW.Rider;
   END IF;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_aggiorna_media_recensioni
AFTER INSERT ON Recensione
FOR EACH ROW
EXECUTE FUNCTION aggiorna_media_recensioni();

--Verifica che il mittente del messaggio sia qualcuno a cui è associato l'ordine
CREATE OR REPLACE FUNCTION verifica_RR_Chat()
RETURNS TRIGGER AS $$
DECLARE
   TipoChat TipoRC;
   IDUtenteR INT;
   IDUtenteCliente INT;
BEGIN
   -- Recupera il tipo di chat
   SELECT Tipo
   INTO TipoChat
   FROM Chat
   WHERE IDChat = NEW.Chat;
```

```
    IF TipoChat = 'Rider' THEN
        -- Recupera il codice utente del rider associato all'ordine a cui è associata la chat
        SELECT r.Utente
        INTO IDUtenteR
        FROM Rider r
        JOIN Ordine O ON O.Rider = r.Codice
        JOIN Chat Ch ON Ch.Ordine = O.IDOrdine
        WHERE Ch.IDChat = NEW.Chat;
    ELSIF TipoChat = 'Ristorante' THEN
        -- Recupera il codice utente del ristoratore che dirige il ristorante a cui è associata la chat a cui è associata
la chat
        SELECT d.Utente
        INTO IDUtenteR
        FROM Dirige d
        JOIN Ordine O ON O.Ristorante = d.Ristorante
        JOIN Chat Ch ON Ch.Ordine = O.IDOrdine
        WHERE Ch.IDChat = NEW.Chat;
    END IF;

    -- Se il mittente NON è uguale al Ristorante/Rider
    IF IDUtenteR IS NOT NULL AND IDUtenteR != NEW.Mittente THEN
        -- Recupera il codice cliente associato all'ordine a cui è associata la chat
        SELECT Cliente
        INTO IDUtenteCliente
        FROM Ordine O
        JOIN Chat Ch ON O.IDOrdine = Ch.Ordine
        WHERE Ch.IDChat = NEW.Chat;

        --Se il cliente non è il mittente lancia una eccezione
        IF IDUtenteCliente IS NOT NULL AND IDUtenteCliente != NEW.Mittente THEN
            RAISE EXCEPTION 'Il Mittente % che hai inserito come mittente del messaggio non è quello associato
all ordine e alla chat, intendevi forse il % % oppure il cliente %',
                NEW.Mittente,TipoChat, IDUtenteR, IDUtenteCliente;
        END IF;
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger
AFTER INSERT ON Messaggio
FOR EACH ROW
EXECUTE FUNCTION verifica_RR_Chat();

CREATE OR REPLACE FUNCTION delete_CSC()
```

```
RETURNS TRIGGER AS $$
BEGIN

   DELETE FROM CSC WHERE Cliente = OLD.Utente;

   DELETE FROM Borsellino WHERE Cliente = OLD.Utente;

   UPDATE ORDINE
   SET Cliente = NULL
   WHERE Cliente = OLD.Utente;

   UPDATE Recensione
   SET Cliente = NULL
   WHERE Cliente = OLD.Utente;

   RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_delete_cliente
BEFORE DELETE ON Cliente
FOR EACH ROW
EXECUTE FUNCTION delete_CSC();

CREATE OR REPLACE FUNCTION delete_Coordinate()
RETURNS TRIGGER AS $$
BEGIN

   DELETE FROM Coordinate WHERE Rider = OLD.Codice;

   RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_delete_rider
BEFORE DELETE ON Rider
FOR EACH ROW
EXECUTE FUNCTION delete_Coordinate();


CREATE OR REPLACE FUNCTION delete_Utente()
RETURNS TRIGGER AS $$
BEGIN

   DELETE FROM Dirige WHERE Utente = OLD.IDUtente;

   DELETE FROM Cliente WHERE Utente = OLD.IDUtente;
```

```
    DELETE FROM Rider WHERE Utente = OLD.IDUtente;

    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_delete_utente
BEFORE DELETE ON Utente
FOR EACH ROW
EXECUTE FUNCTION delete_Utente();

CREATE OR REPLACE FUNCTION delete_Dirige()
RETURNS TRIGGER AS $$
BEGIN

    DELETE FROM Recensione WHERE Ristorante = OLD.Ristorante;

    UPDATE ORDINE
    SET Ristorante = NULL
    WHERE Ristorante = OLD.Ristorante;


    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_delete_dirige
BEFORE DELETE ON Dirige
FOR EACH ROW
EXECUTE FUNCTION delete_Dirige();

CREATE OR REPLACE FUNCTION delete_Ristorante()
RETURNS TRIGGER AS $$
BEGIN

    DELETE FROM Ristorante WHERE IDRistorante = OLD.Ristorante;

    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_delete_ristorante
AFTER DELETE ON Dirige
FOR EACH ROW
EXECUTE FUNCTION delete_Ristorante();
```

```
CREATE OR REPLACE FUNCTION delete_RistoranteB()
RETURNS TRIGGER AS $$
BEGIN

   DELETE FROM Piatto WHERE IDRistorante = OLD.IDRistorante;
   DELETE FROM Lista WHERE Ristorante = OLD.IDRistorante;
   DELETE FROM Appartiene WHERE Ristorante = OLD.IDRistorante;

   RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_delete_RistoranteB
BEFORE DELETE ON Ristorante
FOR EACH ROW
EXECUTE FUNCTION delete_RistoranteB();


CREATE OR REPLACE FUNCTION delete_Piatto()
RETURNS TRIGGER AS $$
BEGIN

   DELETE FROM Contiene WHERE Piatto = OLD.IDPiatto;
   DELETE FROM Include WHERE Piatto = OLD.IDPiatto;
   DELETE FROM Composto WHERE Piatto = OLD.IDPiatto;
   DELETE FROM Raggruppato WHERE Piatto = OLD.IDPiatto;

   RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION update_sconto_trigger()
RETURNS TRIGGER AS $$
BEGIN
   INSERT INTO CodiceSconto(Codice,Valore)
   VALUES ('AUSILIARE', 0);

   UPDATE CSC
   SET Codice = 'AUSILIARE'
   WHERE Codice = OLD.Codice;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trg_update_sconto
BEFORE UPDATE ON CodiceSconto
FOR EACH ROW
EXECUTE FUNCTION update_sconto_trigger();

CREATE OR REPLACE FUNCTION delete_sconto_trigger()
RETURNS TRIGGER AS $$
BEGIN
   UPDATE CSC
   SET Codice = NEW.Codice
   WHERE Codice = 'AUSILIARE';

   DELETE FROM CodiceSconto
   WHERE Codice = 'AUSILIARE';

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_update_scontoB
AFTER UPDATE ON CodiceSconto
FOR EACH ROW
EXECUTE FUNCTION delete_sconto_trigger();
```

Cancellazione a Cascata di un Utente e delle Entità Associate

Cancellazione di un utente che dirige dei ristoranti
```
DELETE FROM Utente
WHERE IDUtente = 1
```

Cancellazione di un utente-Rider
```
DELETE FROM Utente
WHERE IDUtente = 7
```

Cancellazione di un utente-Cliente
```
DELETE FROM Utente
WHERE IDUtente = 10
```

Cambio di Codice
```
UPDATE CodiceSconto
SET Codice = 'SCONTO100',  Valore = 100.00
WHERE Codice = 'DISCOUNT10';
```