

# OpenTable

## - Gestore di Ristoranti Intelligente -

### 1. Raccolta delle specifiche della realtà d'interesse

#### Descrizione

Si vuole progettare una base di dati per la **gestione intelligente di diversi ristoranti**. Ogni ristorante è identificato da un Codice ed è rappresentato tramite un nome. Ogni ristorante può avere più sedi, di cui si conosce l'ID, un indirizzo e la città dove risiede. Ogni attività assume diversi dipendenti, di cui si conoscono un codice identificativo del dipendente, un identificativo dei dati anagrafici (quali nome, cognome, data di nascita), tramite il codice fiscale, un livello che rappresenta i permessi del dipendente nei confronti dell'attività e uno stipendio. Ogni attività possiede un menu, gestito da un dipendente di un certo livello di permessi e responsabilità. Il menu viene rappresentato con un codice identificativo. Ogni tavolo appartenente all'attività visiona il menu e viene identificato tramite un codice, con diversi attributi quali il numero di posti in totale di norma e quelli occupati. Ogni tavolo effettua un'ordinazione che viene identificata tramite un codice ed ha attributi quali il conto complessivo dell'ordinazione e la data d'ordinazione. Essa comprende diversi prodotti ed ogni prodotto ha informazioni quali un codice univoco, un nome ed un costo. Ogni prodotto può essere un cibo o bibita. Ogni cibo ha informazioni come il peso in grammi, il tipo del piatto (primo, secondo, contorno, etc.) e gli ingredienti. Ogni bibita, invece, ha ulteriori informazioni come tasso Alcolemico e capacità in cl. Ogni tavolo effettua diverse recensioni (identificate tramite un codice univoco) ed ogni recensione riguarda un solo prodotto (cibo o bevanda).

#### Specifiche della realtà d'interesse

La realtà che andiamo a rappresentare riguarda la gestione intelligente di diversi ristoranti, attraverso un'idea vincente, quella di poter monitorare in sicurezza le diverse ordinazioni e consentendo al cliente la pura gestione del futuro del ristorante. Per definizione, ogni ristorante basa i suoi profitti totalmente sul pensiero, il sentimento e la frequenza della sua clientela. I ristoranti, però, gestiscono e visionano indirettamente (e non sempre) le reazioni dei clienti, le loro preferenze, i piatti preferiti dalla clientela e gli ingredienti che più vorrebbero in un determinato piatto. Ciò che ha spinto il nostro team a voler adoperarsi per la gestione di ristoranti è stato proprio questo. Rendere attivo e diretto il rapporto tra il ristorante e la clientela, tramite l'utilizzo di monitor accessibili solo dai tavoli che stanno effettuando un'ordinazione e che garantiranno al gestore del ristorante un insieme di informazioni, quali i piatti più ordinati, meno ordinati, recensioni sui prodotti dello chef, piatti preferiti dai clienti e soprattutto possibili pensieri al di fuori dei prodotti, quali un pessimo rapporto qualità/prezzo o consigli per massimizzare la gestione o la qualità dell'attività presa in causa. L'abbiamo vista come idea vincente della nostra gestione **intelligente** delle attività di

ristorazione poiché garantiamo una gestione delle recensioni e del sentimento del cliente. La gestione è onerosa ma proprio queste funzionalità diverse, intuitive e innovative potrebbero essere motivo di discussione e di maggiore flusso di clientela.

È importante sapere che la maggior parte delle entità che rappresentano lo schema concettuale del nostro progetto è stata pensata anche per un maggior livello di sicurezza e di intelligibilità semplice ed efficace con gli utenti (non registrati). L'attività ha diversi dipendenti e solo un dipendente, di un certo livello, può interagire con il menù che verrà monitorato, e modificato eventualmente, ogni settimana. Il tavolo è il concetto principale perché solo il tavolo può visionare il menù, solo il tavolo può effettuare ordinazioni e solo il tavolo, dopo queste ordinazioni, può effettuare le recensioni (in anonimo, senza problemi di giudizio, per essere schietti e diretti). Nessun agente esterno potrà intaccare l'insieme di ordinazioni o recensioni dell'attività e le recensioni verranno giornalmente monitorate.

Come si potrà vedere, il tavolo ha un numero di posti approssimativi ed un numero di posti realmente occupati. Questo per facilitare la gestione della visione dei tavoli liberi per determinate clientele (di un certo numero di persone) e capire quale tavolo ha approssimativamente un numero di posti totali molto vicino a quello desiderato. Il numero dei posti occupati sarà importante per capire se il tavolo è libero ( $\text{numOcc} = 0$ ) o se è occupato e quanti monitor accendere ( $\text{numOcc} > 0$ ).

Tuttavia, la gestione intelligente del nostro team può essere adottata da più attività e quest'ultime possono risiedere in più sedi. Ogni attività ha una gestione di livelli per la responsabilità e permessi sulla sede. Ogni tavolo può effettuare recensioni riguardanti un solo prodotto alla volta, come cibo o bevanda. Ogni prodotto è identificato da due diverse stringhe, una che identifica univocamente quel prodotto in particolare ed uno che rappresenta il tipo, come il nome della soda della bibita, l'Alcolemico, l'acqua (come ad es. Coca-Cola, Pepsi, ... ) e questa scelta progettuale migliorerà e renderà più intuitiva qualsiasi transazione in back-end.

Le modifiche che consigliamo ai proprietari delle attività di ristorazione sono il monitoraggio giornaliero di tutte le recensioni, modifiche da apportare ogni 2 settimane massimo al menù con tanto di visibilità dei piatti e bibite preferiti dai clienti (per poter anche modificarne il costo), massimizzare la qualità dei prodotti recensiti negativamente e migliorare il rapporto qualità/prezzo o gli ingredienti dei prodotti scelti raramente, nel caso. Questa base di dati rappresenta la gestione dell'intelligibilità di tutte le informazioni possibili di ciò che riguarda direttamente e indirettamente il ristorante, nella piena sicurezza software e nella piena coerenza dei dati al livello fisico.

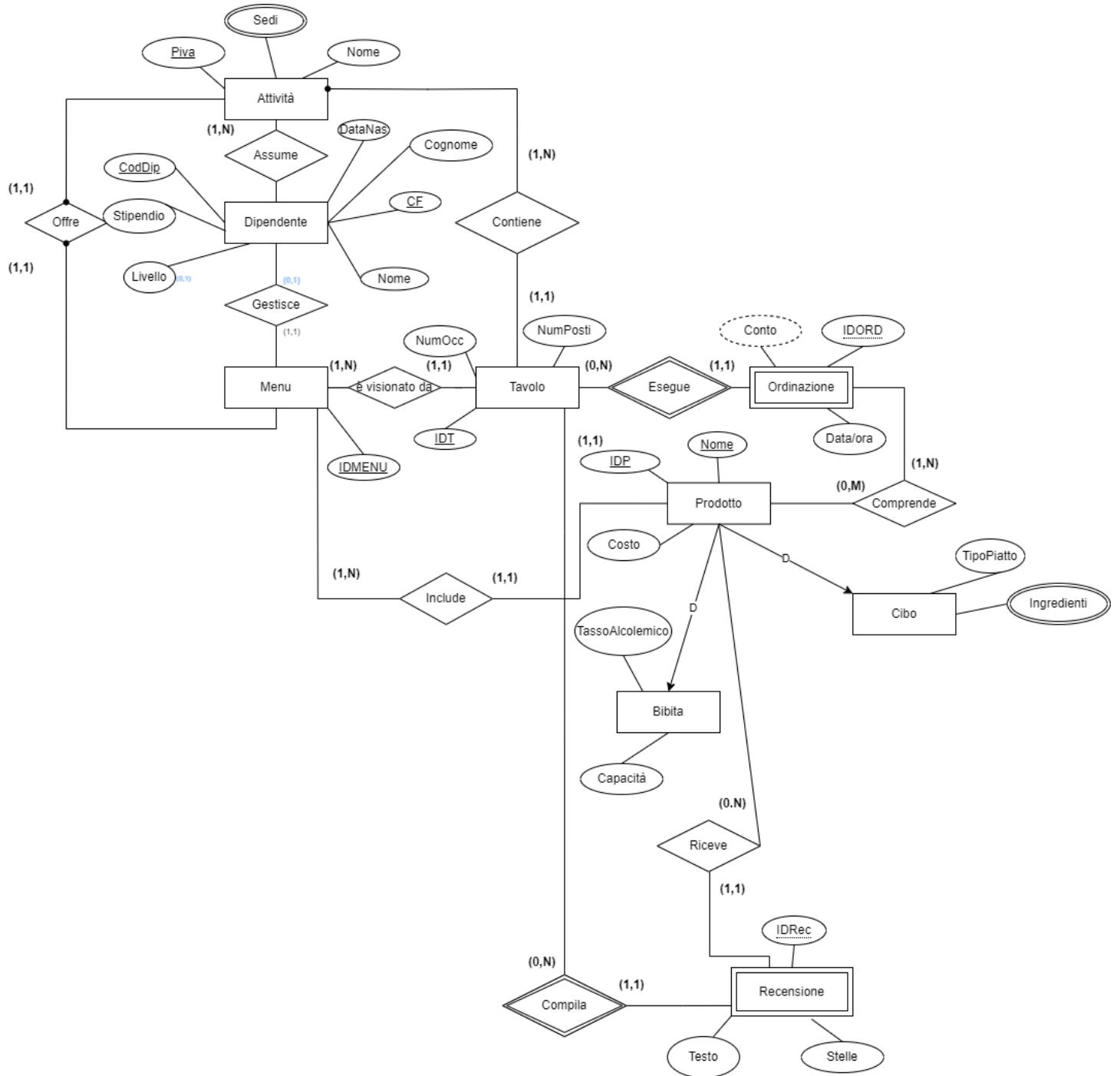
## Glossario dei termini

Termine	Significato
Attività	Attività di ristorazione soggetta alla gestione intelligente della base di dati.
Dipendente	Persona che lavora in una determinata attività di ristorazione.
Menu	Insieme di prodotti che comprende la vasta scelta delle ordinazioni dei clienti nei pubblici esercizi per la consumazione.
Tavolo	Tavolo dell'attività di ristorazione che comprende monitor per l'esecuzione o modifica di ordinazioni e recensioni dei diversi prodotti.
Ordinazione	Insieme di prodotti richiesti dalla clientela seduta al tavolo.
Prodotto	Oggetto delle ordinazioni e consumazioni della clientela.
Cibo	Tipo di prodotto ordinabile e recensibile, contenente diversi ingredienti.
Bibita	Tipo di prodotto ordinabile e recensibile, con tasso Alcolemico maggiore o uguale di 0.
Recensione	Una valutazione (tramite testo facoltativo e stelle) interattiva effettuata dal cliente per un tipo di prodotto.

## 2. Progettazione concettuale della base di dati

### Schema EER Rappresentativo:

222.



## Dizionario delle entità

**Legenda:** sotto-entità, attributo multivалore, attributo ridondante, entità debole, chiave candidata

Entità	Descrizione	Attributi	Identificatore
<b>Attività</b>	Attività commerciale gestita dalla base di dati.	- Piva - Nome - Sedi	Piva
<b>Dipendente</b>	Persona che lavora all'interno dell'attività	- CodDip - CF - Nome - Cognome - DataNas - Livello - Stipendio	CodDip, CF
<b>Menu</b>	Insieme di cibi e bevande visibili e ordinabili	- IDMEN	IDMEN
<b>Tavolo</b>	Tavolo dell'attività che effettua recensioni ed ordinazioni	- IDT - NumOcc - NumPosti	IDT
<b>Ordinazione</b>	Insieme di prodotti desiderati da un determinato tavolo ad una certa data/ora	- IDORD - Data/ora - Conto	IDORD
<b>Prodotto</b>	Oggetto di ordinazioni consumazioni e recensioni di tipo "Cibo" o "Bevanda"	- IDP - Nome - Costo	IDP

<b>Bibita</b>	Tipo di prodotto con una capacità ed un tasso Alcolemico	- Capacità - TassoAlcolemico	/
<b>Recensione</b>	Valutazione effettuata dal cliente riguardante un prodotto	- IDRec - Testo - Stelle	IDRec

### Dizionario delle relazioni

<b>Relazione</b>	<b>Descrizione</b>	<b>Entità coinvolte</b>	<b>Attributi</b>
<b>Assume</b>	Ogni attività assume uno o più dipendenti	Attività(1, N) Dipendente(1, 1)	/
<b>Offre</b>	Ogni attività offre un solo menù	Attività (1,1) Menù (1,1)	/
<b>Contiene</b>	Ogni attività contiene uno o più tavoli	Attività (1, N) Tavolo (1, 1)	/
<b>è visionato da</b>	Un menù è visionato da uno o più tavoli	Menù (1, N) Tavolo (1,1)	/
<b>Gestisce</b>	Un menù è visionato da uno o più dipendenti	Menù (1,N) Dipendente (0,1)	/
<b>Include</b>	Un menù include uno o più prodotti	Menù (1, N) Prodotto (1, 1)	/
<b>Esegue</b>	Ogni tavolo può eseguire una o più ordinazioni	Tavolo (0, N) Ordinazione (1, 1)	/
<b>Comprende</b>	Un'ordinazione comprende uno o più prodotti	Ordinazione (1, N) Prodotto (0, M)	/
<b>Compila</b>	Ogni tavolo compila delle recensioni su singoli prodotti	Tavolo (0, N) Recensione (1, 1)	/

<b>Riceve</b>	Ogni prodotto può ricevere delle recensioni	Prodotto (0,N) Recensione (1,1)	/
---------------	---	------------------------------------	---

### Vincoli non esprimibili nello schema

Oltre ciò che è deducibile dallo schema EER, si tenga conto dei seguenti **vincoli**:

- Il menù è formato dalle sezioni “Preferiti dai clienti”, “Consigliati dallo Chef” (ognuna con massimo 3 portate) e il resto del menù, includendo piatti e bevande. Le prime due sezioni contengono i piatti del resto del menù con maggiore valutazione (o numero di ordinazioni indicate) o preferiti dallo chef. Se, al momento dell’aggiornamento del menù, il piatto desiderato è già presente nel menù, viene spostato nella seconda sezione.. Altrimenti verrà prima creato e poi direttamente inserito (dal proprietario), per la prima volta, all’interno della sezione dello chef.
- Per l’entità “Recensione” l’inserimento di un testo è facoltativo, mentre delle stelle è obbligatorio.
- Per l’entità “Tavolo”, l’attributo “NumPosti” non può essere minore di 2.
- Per l’entità Dipendente, l’attributo “Livello” è compreso tra 1 e 3. Il livello 3 è assegnato al proprietario e consente il massimo dei permessi, mentre il 2 rappresenta la possibilità di modificare solo ed esclusivamente la sezione del menù “Consigliati dallo chef”. Il livello 1 sarà per il cassiere ed il cameriere. Il dipendente non può avere meno di 16 anni.

## 3. Definizione delle procedure per la gestione della base di dati

### Tavola dei volumi

Definiamo di seguito la tavola dei volumi della base di dati.

Concetto	Tipo	Carico Applicativo
Attività	E	2
Dipendente	E	30
Menù	E	2
Tavolo	E	40
Ordinazione	E	160
Prodotto	E	120

<b>Cibo</b>	E	80
<b>Bibita</b>	E	40
<b>Recensione</b>	E	120
<b>Assume</b>	R	30
<b>Offre</b>	R	2
<b>Contiene</b>	R	40
<b>è visionato da</b>	R	40
<b>Gestisce</b>	R	4
<b>Include</b>	R	120
<b>Esegue</b>	R	160
<b>Comprende</b>	R	960
<b>Compila</b>	R	120
<b>Riceve</b>	R	120

## Tavola delle operazioni

Definiamo di seguito la tavola delle operazioni per la gestione dei dati memorizzati nella base di dati.

	<b>Operazione</b>	<b>Tipo</b>	<b>Frequenza</b>
1	Creare una nuova attività	I	2/aa
2	Assumere un dipendente	I	30/aa
3	Creare il menù	I	2/aa
4	Aggiornare il menù	I	8/mm
5	Creazione nuovo prodotto	I	2/mm
6	Registrazione di un'ordinazione	I	160/gg
7	Stampa i conti di tutte le ordinazioni giornaliere	B	2/gg
8	Rendi occupato un tavolo	I	160/gg

9	Registra recensione	I	120/gg
10	Stampa i prodotti con recensione più alta	B	8/mm
11	Stampa i prodotti	B	480/mm
12	Stampa i prodotti più ordinati	I	8/mm
13	Visualizza gli ingredienti dei prodotti	I	480/mm

## 4. Progettazione logica

### Analisi delle ridondanze

Il dato ridondante è l'attributo “Conto” dell'entità Ordinazione. Infatti, sarebbe possibile avere la cifra totale delle ordinazioni, (Singole, giornaliere o mensili), attraverso un'interrogazione su questo dato. Supponendo che l'attributo abbia un peso di 4 byte, essendo un normale float, e considerato che il volume dell'entità Ordinazione è uguale a 160, il dato andrebbe ad occupare uno spazio totale di circa **640 byte**. Per decidere se mantenere o meno il dato ridondante è necessario calcolare, per le operazioni che lo coinvolgono, la differenza nel numero di accessi con e senza quest'ultimo.

### Tavola degli accessi

#### Operazione 7

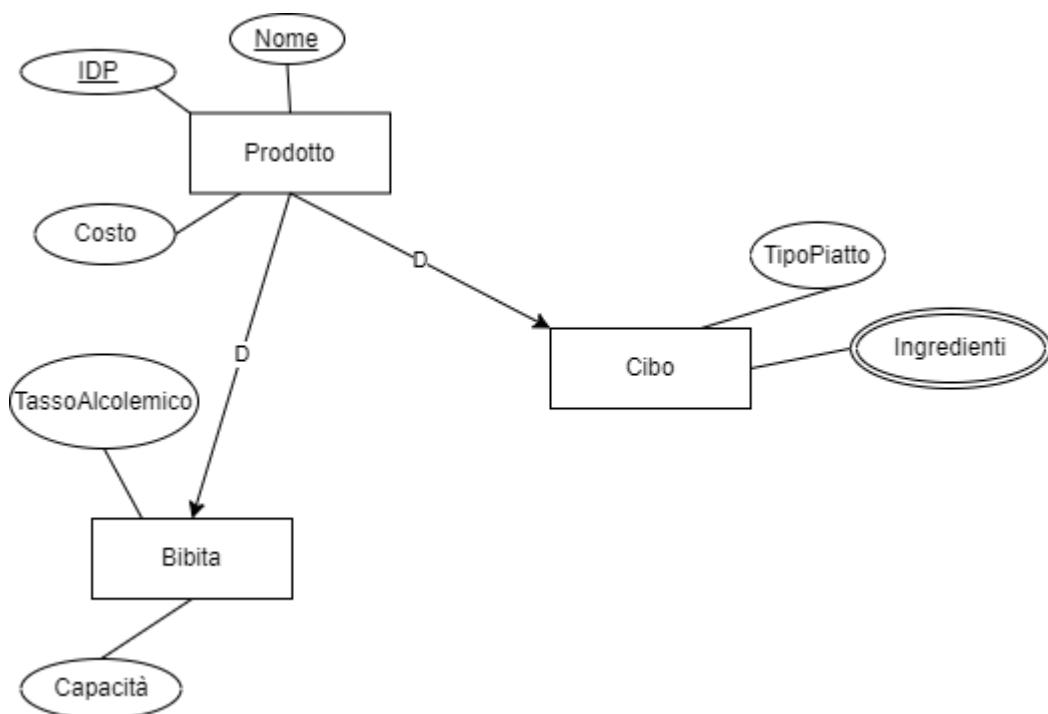
	Calcolo con ridondanza				Calcolo senza ridondanza			
	Tabella	Tipo	Accessi	Tipo accessi	Tabella	Tipo	Accessi	Tipo accessi
Ordinazione	E	80	L		Ordinazione	E	80	L
					Comprende	R	480	L
					Prodotto	E	480	L
<b>Totale</b>	$80 \times 2/\text{gg} = \mathbf{160 \text{ a/gg}} = \mathbf{4800 \text{ a/mm}}$				$(80 + 480 + 480) \times 2/\text{gg} = \mathbf{2080 \text{ a/gg}} = \mathbf{62400 \text{ a/mm}}$			

Totale accessi con ridondanza	Totale accessi senza ridondanza
<b>4800 a/mm + 640 byte</b>	<b>62400 a/mm</b>

Grazie alla ridondanza dell'attributo “Conto” presente tra le informazioni di ogni ordinazione, abbiamo un numero estremamente inferiore di accessi per effettuare una delle operazioni più frequenti e importanti dell'attività commerciale. L'operazione 7 è l'unica che presenta la richiesta della somma totale giornaliera delle ordinazioni. Pertanto, 4800 accessi al mese con 640 byte **sono preferibili** rispetto a 62400 accessi al mese senza ridondanza.

### Eliminazione delle gerarchie

Nello schema inizialmente elaborato, è presente la seguente specializzazione dell'entità “Prodotto”:

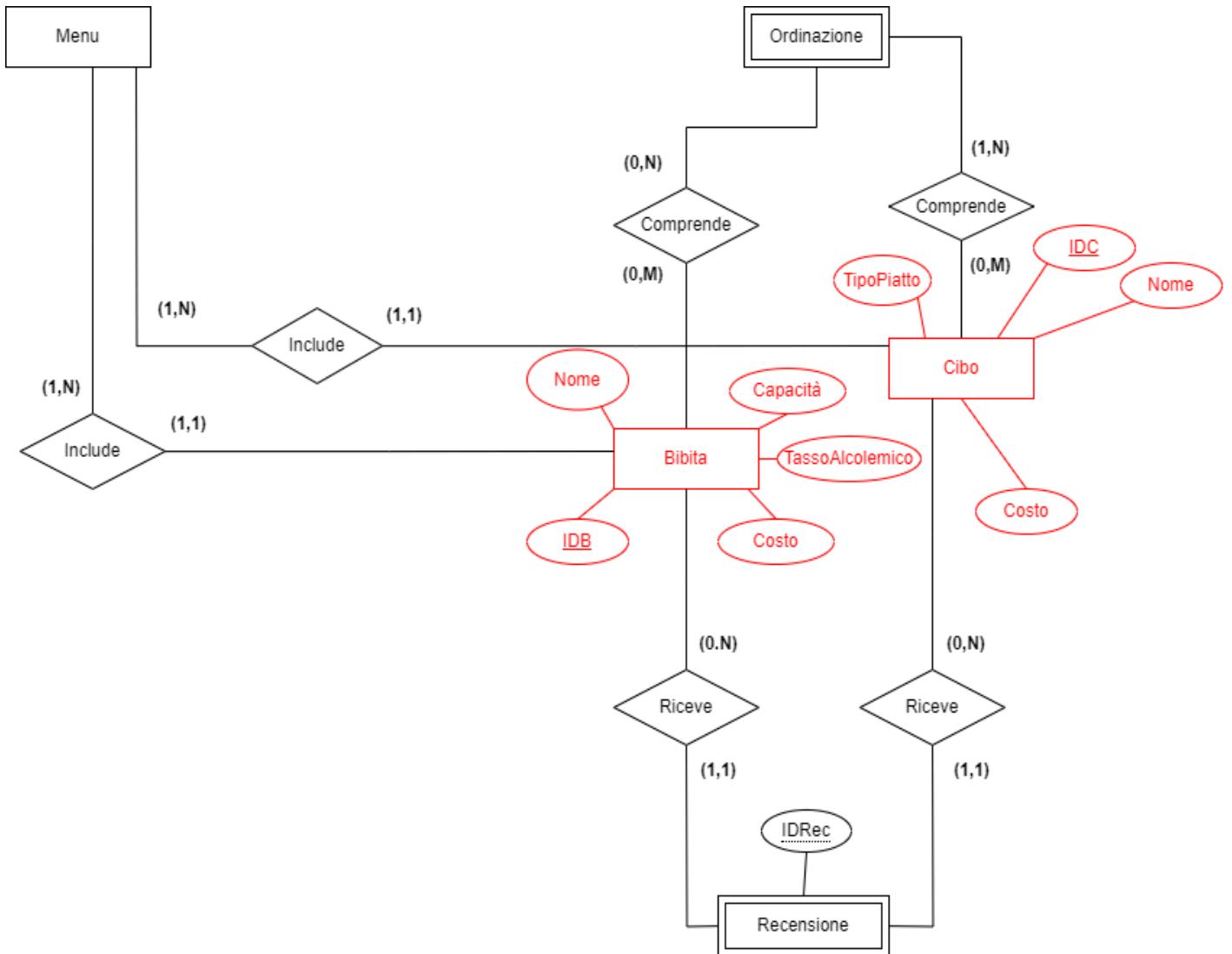


In questa fase di progettazione logica, è necessario individuare un metodo efficace di ristrutturazione che permetta l'eliminazione di questa gerarchia.

La scelta effettuata è quella di eliminare la generalizzazione “Prodotto” e far diventare due vere e proprie entità, con codice univoco, le specializzazioni “Cibo” e “Prodotto”. I valori NULL, purtroppo, peggiorano le operazioni, rendendole maggiormente onerose a causa dei vari controlli obbligatori da effettuare. In altri casi, avremo dovuto controllare valori null e recensioni inutilizzate per definire lo stesso concetto. Date le operazioni interessate dalla

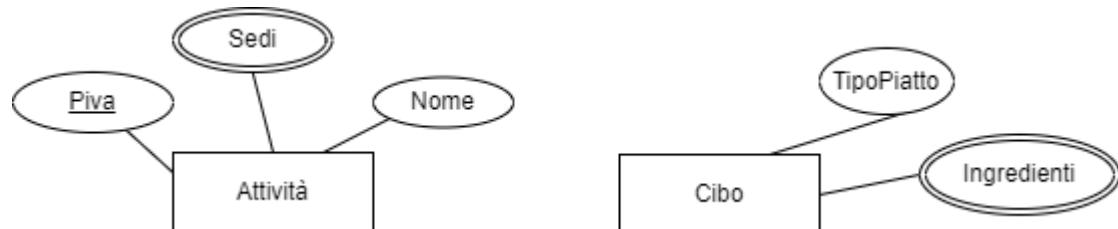
gestione della nostra base di dati, abbiamo deciso di distinguere completamente le due entità.

Le relazioni e le entità sono state modificate in questo modo:

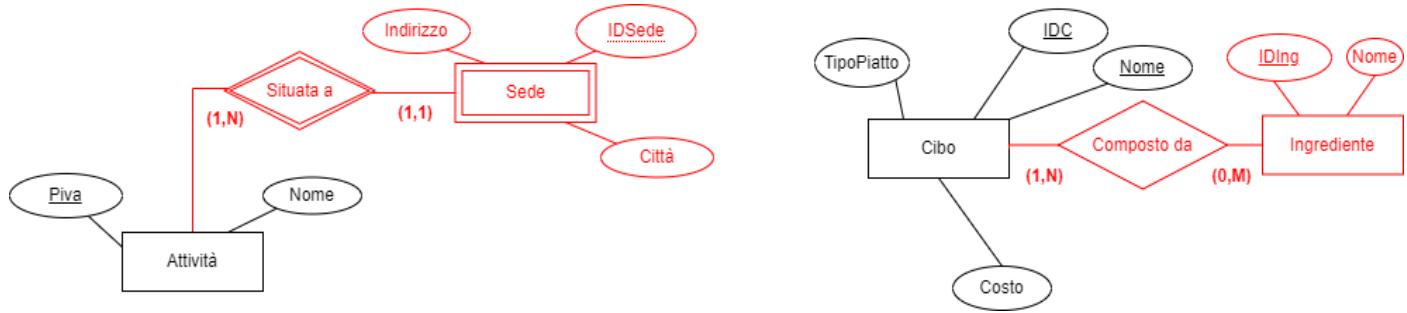


## Eliminazione dell'attributo multivaleore

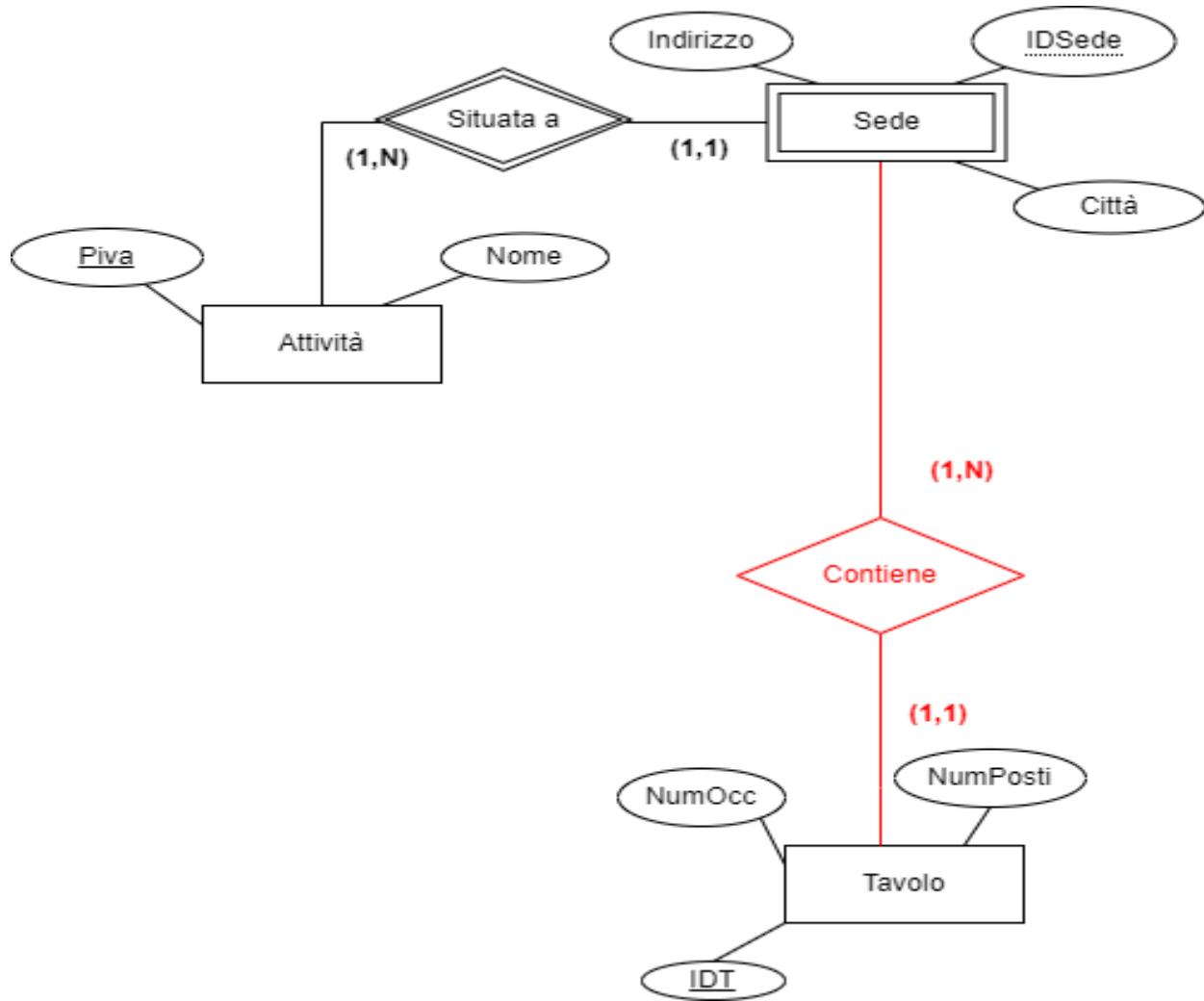
Nello schema inizialmente elaborato, compaiono due attributi multivaleore: Tale forma di attributo va risolto in maniera differente in fase di progettazione logica.



Si sceglie quindi di definire due nuove entità: l'entità debole “Sede” che sarà in relazione con “Attività” e l'entità “Ingrediente” che sarà in relazione con “Cibo”.

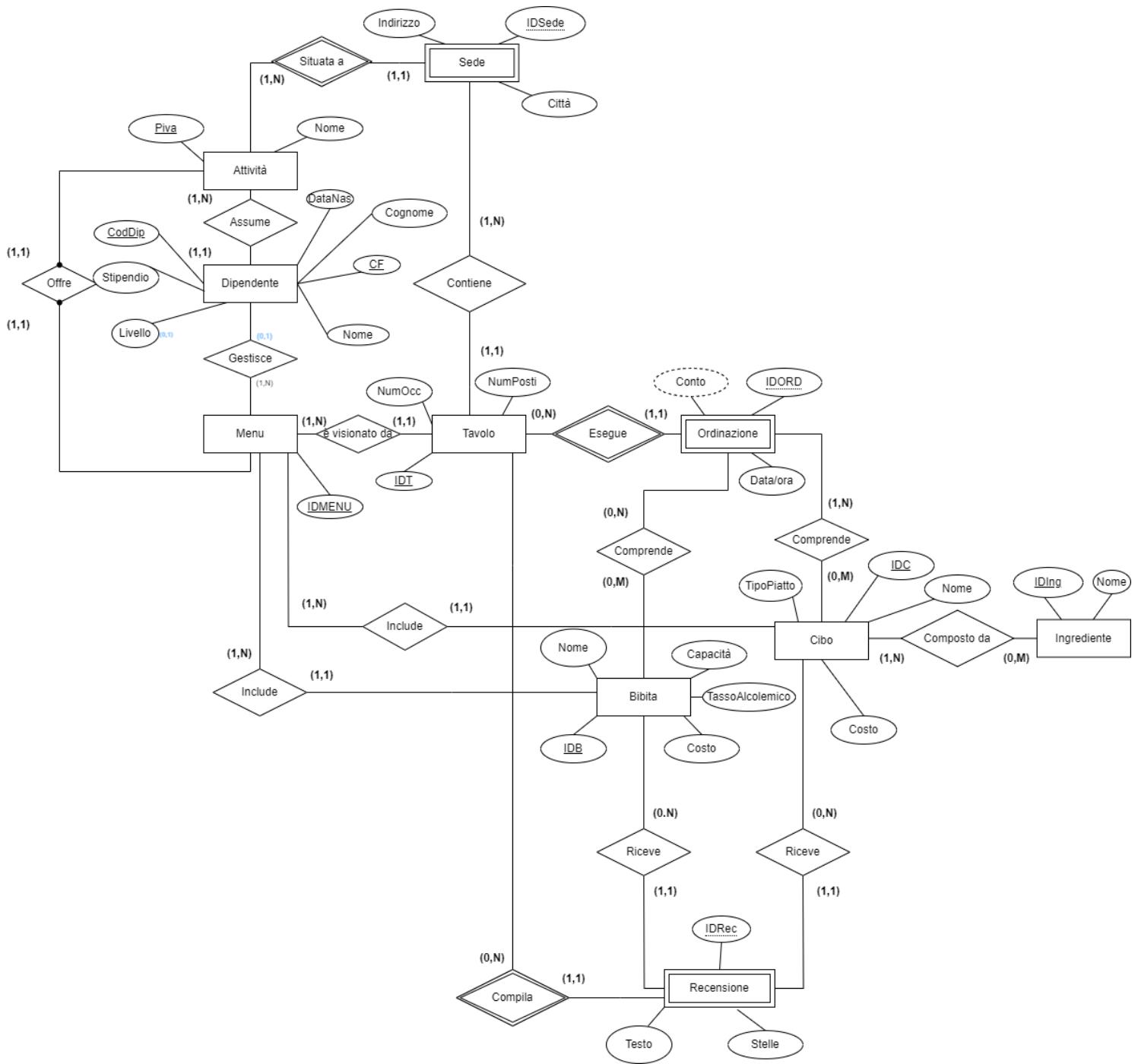


Data la creazione dell'entità debole “Sede”, si evince che i tavoli, in realtà, appartengono ad una singola sede e non all'attività commerciale (concetto astratto rispetto alla sede, struttura fisica). A questo punto modifichiamo la relazione tra “Attività” e “Tavolo”:



### Schema EER ristrutturato

Al termine della fase di ristrutturazione, lo schema EER completo che ne deriva è il seguente:



## Schema relazionale

Si procede al mapping della base di dati:

**Attività**(Piva, nome, Menù↑)

**Sede**(IDSede, Piva↑, città, indirizzo)

**Dipendente**(CF, CodDip, nome, cognome, dataNas, stipendio, livello, Attività↑)

**Gestisce**(Menu↑, Dipendente↑)

**Menu**(IDMENU)

**Tavolo**(IDT, numOcc, numPosti, Sede↑, Attività↑, Menu↑)

**Ordinazione**(Tavolo↑, IDORD, conto(derivato), data/ora)

**Cibo**(IDC, Nome, tipoPiatto, costo, Menu↑)

**Ingrediente**(IDIng, nome)

**CompostoDa**(Cibo↑, Ingrediente↑)

**Comprende**(Ordinazione↑, Cibo↑)

**Bibita**(IDB, Nome, capacità, tassoAlcolemico, costo, Menu↑)

**Comprende**(Ordinazione↑, Bibita↑)

**Recensione**(Tavolo↑, IDRec, testo, stelle, Bibita↑, Cibo↑)

## DF e Normalizzazione

Prima di analizzare la forma della normalizzazione del seguente progetto, è stato deciso di riportare (e di correggere, nel caso) le dipendenze funzionali coinvolte. Per ogni relazione, si analizzeranno le specifiche della realtà rappresentata e i vincoli.

- **Attività**: è una relazione formata da tre attributi: Piva (PK), nome e codice identificativo del menù utilizzato. Questi ultimi dipendono funzionalmente da Piva ed è, per questo motivo, una dipendenza **piena**.
  - FD: PK -> {Nome, Menù}
- **Sede**: è una relazione formata da una coppia di attributi come Primary Key e gli attributi indirizzo e città. Questi attributi dipendono funzionalmente dalla locazione dell'azienda e l'azienda stessa che richiede una sede per svolgere le sue attività di ristorazione. È dunque una dipendenza funzionale **piena**.
  - FD: {Piva, IDSede} -> {indirizzo, città}
- **Dipendente**: è una relazione formata da diversi attributi che rappresentano informazioni anagrafici e di lavoro. Tuttavia, proprio per questo motivo, un sottoinsieme di attributi sarà dipendente funzionalmente solo dal codice fiscale, mentre un altro sottoinsieme di attributi sarà dipendente solo ed esclusivamente dal codice del dipendente. Per renderlo concettualmente e formalmente giusto, avremo bisogno di una nuova relazione per differenziare i dati del dipendente come lavoratore e dati del dipendente come persona. Le modifiche saranno:
  - **Dipendente**(CodDip, livello, stipendio, Attività↑, CodAnagrafica↑)
  - **Anagrafica**(CF, nome, cognome, dataNas)
  - Dipendenza funzionale prima: FD: {CodDip -> {livello, stipendio, Attività}, CF -> {nome, cognome, dataNas}}
  - Dipendenze funzionali ora: FDDipendente: CodDip -> {livello, stipendio, Attività, CodAnagrafica}, FDAnagrafica: CF -> {nome, cognome, dataNas}
  - Aggiungendo la relazione "Anagrafica", avremo delle dipendenze funzionali **piene** e non parziali.
- **Gestisce**: è una relazione priva di attributi non chiave. Pertanto, non è necessario lo sviluppo o lo studio delle sue dipendenze funzionali.
- **Menù**: è una relazione che rappresenta il menù di una determinata attività. Menù ha come unica informazione il suo codice identificativo. Pertanto, non è

utile lo studio della sua dipendenza funzionale.

- **Tavolo:** è una relazione che rappresenta il tavolo fisico che comprende dei monitor in grado di effettuare ordinazioni, visionare il menù e compilare delle piccole recensioni. Il tavolo ha diverse informazioni quali il numero di posti occupati, numero di posti approssimativi, la sede in cui è posta, l'attività della sede ed il menù che può visionare. Queste informazioni dipendono funzionalmente dal codice identificativo univoco del tavolo. Vi è una dipendenza funzionale **piena**: FD: IDT -> {numOcc, numPosti, Sede, Attività, Menù}
- **Ordinazione:** L'ordinazione è la relazione che rappresenta le diverse richieste della clientela presente al tavolo. Gli attributi sono diversi: il codice identificativo del tavolo che ha ordinato, il codice dell'ordinazione, il conto e la data/ora. La coppia di attributi che forma le primary key crea una dipendenza funzionale **piena** poichè entrambi i codici rappresentano tutte le possibili informazioni dell'ordinazione. FD: {IDT, IDORD} -> {conto, data/ora}.
- **Cibo:** Cibo è una relazione che rappresenta un tipo di prodotto. Ogni cibo ha diverse informazioni: il tipo del piatto ed il costo. Esso viene identificato univocamente tramite l'attributo "IDC" (codice). La dipendenza funzionale è **piena**. FD: {IDC} -> {Nome, tipoPiatto, costo}.
- **Ingrediente:** l'ingrediente compone le portate dell'attività. Esso ha un codice identificativo ed un nome e da ciò si evince una dipendenza funzionale **piena** banale. FD: IDIng -> nome.
- **CompostoDa:** è una relazione che contiene abbastanza informazioni per collegare un certo ingrediente ad una pietanza. Le uniche informazioni rivedono una coppia di primary key. Pertanto, non è utile la visione della dipendenza funzionale.
- **Comprende:** è una relazione che rivede le relazioni presenti tra una certa ordinazione e le pietanze desiderate. Tuttavia, essendo una relazione avente, come uniche informazioni, una coppia di primary key, è considerabile banalmente inutile lo studio delle dipendenze funzionali.
- **Bibita:** Bibita, come "Cibo", è una relazione con la stessa logica per l'identificazione univoca del prodotto. Le uniche differenze sono nel resto degli attributi (non chiave), quali capacità, tassoAlcolemico e il costo (presente anche in Cibo). La logica delle dipendenze funzionali è la stessa e sarà **piena**.
- **Recensione:** è una relazione che rivede tutte le informazioni compilate da un tavolo riguardo un prodotto ordinato. Gli attributi chiave sono il Tavolo che compila la recensione ed il codice identificativo univoco di quest'ultima, mentre quelli non chiave sono Stelle, Testo, l'eventuale Bibita recensita o l'eventuale Cibo recensito. Tra i vincoli evidenziati in precedenza, si può notare come sia possibile compilare Recensione con un solo tipo di prodotto alla volta. Analizzando i vari vincoli, notiamo che abbiamo una dipendenza funzionale **piena**, poiché, per definizione del codice identificativo univoco IDRec, si hanno tutte le informazioni sul testo e le stelle utilizzate nella compilazione del tavolo

il quale ha voluto esprimere una valutazione su un prodotto provato (Bibita, Cibo). FD: {Tavolo, IDRec} -> {testo, stelle, Bibita, Cibo}.

Lo schema è in **1FN** perché tutti gli attributi sono stati designati in modo tale da avere informazioni atomiche, garantendo maggiore coerenza e coesione tra i dati. Lo schema è in **2FN**, come si evince dallo studio delle dipendenze funzionali piene effettuato poco prima. Lo schema è in **3FN** perché le dipendenze non sono mai transitive.

## Il modello logico revisionato dallo studio delle dipendenze funzionali sarà il seguente:

**Attività**(Piva, nome, Menù↑)

**Sede**(IDSede, Piva↑, città, indirizzo)

**Dipendente**(CodDip, stipendio, livello, Anagrafica↑, Attività↑)

**Anagrafica**(CE, nome, cognome, dataNas)

**Gestisce**(Menu↑, Dipendente↑)

**Menu**(IDMENU)

**Tavolo**(IDT, numOcc, numPosti, Sede↑, Attività↑, Menu↑)

**Ordinazione**(Tavolo↑, IDORD, conto(derivato), data/ora)

**Cibo**(IDC, Nome, tipoPiatto, costo, Menu↑)

**Ingrediente**(IDIng, nome)

**CompostoDa**(Cibo↑, Ingrediente↑)

**Comprende**(Ordinazione↑, Cibo↑)

**Bibita**(IDB, Nome, capacità, tassoAlcolemico, costo, Menu↑)

**Comprende**(Ordinazione↑, Bibita↑)

**Recensione**(Tavolo↑, IDRec, testo, stelle, Bibita↑, Cibo↑)

## Vincoli non esprimibili nello schema (Finalizzato)

Oltre ciò che è deducibile dallo schema EER, si tenga conto dei seguenti **vincoli**:

- Il **menù** è formato dalle sezioni “Preferiti dai clienti”, “Consigliati dallo Chef” (ognuna con massimo 3 portate) e il resto del menù, includendo piatti e bevande. Le prime due sezioni contengono i piatti del resto del menù con maggiore valutazione (o numero di ordinazioni indicate) o preferiti dallo chef. Se, al momento dell’aggiornamento del menù, il piatto desiderato è già presente nel menù, viene spostato nella seconda sezione.. Altrimenti verrà prima creato e poi direttamente inserito (dal proprietario), per la prima volta, all’interno della sezione dello chef.
- Per “**Recensione**” l’inserimento di un testo è facoltativo, mentre delle stelle è obbligatorio. Solo un prodotto tra cibo e prodotto può essere compilato, mentre l’altro verrà inserito come *null*
- Per “**Tavolo**”, l’attributo “NumPosti” non può essere minore di 2. grazie a

numOcc si può vedere se un tavolo è libero o meno (numOcc == 0 -> libero; numOcc > 0 -> occupato).

- Per **Gestisce**, il codice del dipendente si deve riferire ad un lavoratore con un livello maggiore o uguale di due, altrimenti la gestione è negata.
- Per la relazione **Dipendente**, l'attributo "Livello" è compreso tra 1 e 3. Il livello 3 è assegnato al proprietario e consente il massimo dei permessi, mentre il 2 rappresenta la possibilità di modificare solo ed esclusivamente la sezione del menù "Consigliati dallo chef". Il livello 1 sarà per il cassiere ed il cameriere. Il dipendente non può avere meno di 16 anni.
- Per la relazione **Bibita**, l'attributo tassoAlcolemico ha come valore 0 se è una bevanda analcolica, altrimenti alcolica.

## 5. SQL:

### Creazione Tabelle

```
DROP DATABASE IF EXISTS LocaleInterattivo;  
CREATE DATABASE LocaleInterattivo;  
USE LocaleInterattivo;
```

```
CREATE TABLE Anagrafica(  
    CF CHAR(16) NOT NULL,  
    Nome VARCHAR(20),  
    Cognome VARCHAR(20),  
    DataNas DATE,  
    PRIMARY KEY (CF)  
)
```

```
CREATE TABLE Menu(  
    IDMenu VARCHAR(10) NOT NULL,  
    PRIMARY KEY (IDMenu)  
)
```

```
CREATE TABLE Attività(  
    Piva VARCHAR(11) NOT NULL,  
    Nome VARCHAR(20),  
    IDMenu VARCHAR(10) NOT NULL,  
    PRIMARY KEY (Piva),  
    FOREIGN KEY (IDMenu) REFERENCES Menu(IDMenu)  
)
```

```
CREATE TABLE Dipendente(  
    CodDip CHAR(10) NOT NULL,
```

```
        Stipendio DOUBLE,  
        Livello INT CHECK(Livello >= 1 AND Livello <=3),  
        CF CHAR(16) NOT NULL,  
        Piva VARCHAR(11) NOT NULL,  
        PRIMARY KEY (CodDip),  
        FOREIGN KEY (CF) REFERENCES Anagrafica(CF),  
        FOREIGN KEY (Piva) REFERENCES Attività(Piva)  
);
```

```
CREATE TABLE Sede(  
    IDSede CHAR(10) NOT NULL,  
    Indirizzo VARCHAR(50),  
    Città VARCHAR(25),  
    Piva VARCHAR(11) NOT NULL,  
    PRIMARY KEY (IDSede),  
    FOREIGN KEY (Piva) REFERENCES Attività(Piva)  
);
```

```
CREATE TABLE Tavolo(  
    IDT VARCHAR(2) NOT NULL,  
    NumPosti INT NOT NULL,  
    NumOcc INT NOT NULL,  
    IDSede CHAR(10) NOT NULL,  
    IDMenu VARCHAR(10) NOT NULL,  
    Piva VARCHAR(11) NOT NULL,  
    PRIMARY KEY (IDT),  
    FOREIGN KEY (IDMenu) REFERENCES Menu(IDMenu),  
    FOREIGN KEY (Piva) REFERENCES Attività(Piva),  
    FOREIGN KEY (IDSede) REFERENCES Sede(IDSede)  
);
```

```
CREATE TABLE Ordinazione(  
    IDORD VARCHAR(7) NOT NULL,  
    Dataora DATETIME NOT NULL,  
    Conto DOUBLE NOT NULL,  
    IDT VARCHAR(2) NOT NULL,  
    PRIMARY KEY (IDORD),  
    FOREIGN KEY (IDT) REFERENCES Tavolo(IDT)  
);
```

```
CREATE TABLE Bibita(  
    IDB CHAR(5) NOT NULL,
```

```
Nome VARCHAR(15) NOT NULL,  
Capacità FLOAT NOT NULL,  
TassoAlcolemico FLOAT,  
Costo DOUBLE,  
IDMenu VARCHAR(10) NOT NULL,  
PRIMARY KEY (IDB),  
FOREIGN KEY (IDMenu) REFERENCES Menu(IDMenu)  
);
```

```
CREATE TABLE Cibo(  
IDC CHAR(5) NOT NULL,  
Nome VARCHAR(15) NOT NULL,  
TipoPiatto VARCHAR(25) NOT NULL,  
Costo DOUBLE,  
IDMenu VARCHAR(10) NOT NULL,  
PRIMARY KEY (IDC),  
FOREIGN KEY (IDMenu) REFERENCES Menu(IDMenu)  
);
```

```
CREATE TABLE Ingrediente(  
IDIn CHAR(8) NOT NULL,  
Nome VARCHAR(35) NOT NULL,  
PRIMARY KEY (IDIn)  
);
```

```
CREATE TABLE CompostoDa(  
IDC CHAR(5) NOT NULL,  
IDIn CHAR(8) NOT NULL,  
PRIMARY KEY (IDC, IDIn),  
FOREIGN KEY (IDC) REFERENCES Cibo(IDC),  
FOREIGN KEY (IDIn) REFERENCES Ingrediente(IDIn)  
);
```

```
CREATE TABLE ComprendeC(  
IDC CHAR(5) NOT NULL,  
IDORD VARCHAR(7) NOT NULL,  
PRIMARY KEY (IDC, IDORD),  
FOREIGN KEY (IDC) REFERENCES Cibo(IDC),  
FOREIGN KEY (IDORD) REFERENCES Ordinazione(IDORD)  
);
```

```
CREATE TABLE ComprendeB(
```

```
IDB CHAR(5) NOT NULL,  
IDORD VARCHAR(7) NOT NULL,  
PRIMARY KEY (IDB,IDORD),  
FOREIGN KEY (IDB) REFERENCES Bibita(IDB),  
FOREIGN KEY (IDORD) REFERENCES Ordinazione(IDORD)  
);
```

```
CREATE TABLE Recensione(  
IDRec CHAR(7) NOT NULL,  
Testo VARCHAR (1000),  
Stelle INT CHECK (Stelle>=0 AND Stelle<=5),  
IDB CHAR(5),  
IDC CHAR(5),  
IDT VARCHAR(2) NOT NULL,  
PRIMARY KEY (IDRec),  
FOREIGN KEY (IDB) REFERENCES Bibita(IDB),  
FOREIGN KEY (IDC) REFERENCES Cibo(IDC),  
FOREIGN KEY (IDT) REFERENCES Tavolo(IDT),  
CHECK ((IDC IS NULL AND IDB IS NOT NULL) OR (IDC IS NOT NULL AND IDB  
IS NULL))  
);
```

```
CREATE TABLE Gestisce(  
IDMenu VARCHAR(10) NOT NULL,  
CodDip CHAR(10) NOT NULL,  
PRIMARY KEY (CodDip, IDMenu),  
FOREIGN KEY (CodDip) REFERENCES Dipendente(CodDip),  
FOREIGN KEY (IDMenu) REFERENCES Menu(IDMenu)  
);  
ALTER TABLE Tavolo  
ADD CONSTRAINT tavolo_chk_1 CHECK (NumOcc <= NumPosti AND NumOcc >=  
0 AND NumPosti>=2);
```

## Operazioni

--Operazione 1

```
INSERT INTO Menu  
VALUES(?);
```

```
INSERT INTO Attività  
VALUES(?, ?, ?);
```

-Operazione 2

```
INSERT INTO Anagrafica  
Values(?, ?, ?, ?);
```

```
INSERT INTO Dipendente  
VALUES(?, ?, ?, ?, ?);
```

--Operazione 3

```
INSERT INTO Menu  
VALUES(?);
```

--Operazione 4

--Operazione non fattibile non avendo un campo che identifica se  
--è o non è consigliato dallo chef per poterlo inserire in cima  
--al menù, pertanto attualmente rimane un'operazione visiva al  
--livello del jdbc

--Operazione 5

--Caso Bibita

--\* la capacità è espressa in int, pertanto il numero inserito  
--varrà come cl

```
INSERT INTO Bibita
```

```
Values(?, ?, ?, ?, ?, ?, ?);
```

--Caso Cibo

```
INSERT INTO Cibo
```

```
Values(?, ?, ?, ?, ?, ?, ?);
```

```
INSERT INTO Ingrediente
```

```
Values(?, ?),  
(?, ?),  
(?, ?);
```

```
INSERT INTO CompostoDa
```

```
Values(?, ?),  
(?, ?),  
(?, ?),
```

(?, ?);

--Operazione 6

```
INSERT INTO Ordinazione  
Values(?, ?, ?, ?, ?, ?);
```

```
INSERT INTO ComprendeB  
Values(?, ?);
```

```
INSERT INTO ComprendeC  
Values(?, ?);
```

--Operazione 7

```
SELECT Conto  
FROM Ordinazione  
WHERE YEAR(Dataora) = anno AND MONTH(Dataora) = mese DAY(Dataora) =  
giorno
```

--Operazione 8

--in questo esempio metto solo 1 persona anche perchè un tavolo  
--con una sola persona risulta comunque occupato

```
UPDATE Tavolo  
SET NumOcc = 1  
WHERE IDT = 'Tavolodaoccupare'
```

--Operazione 9

--Caso Cibo

```
INSERT INTO Recensione(IDRc,Testo,Stelle, IDC, IDT)  
Values(?, ?, ?, ?, ?, ?);
```

--Caso Bibita

```
INSERT INTO Recensione(IDRc,Testo,Stelle,>IDB, IDT)  
Values(?, ?, ?, ?, ?, ?);
```

--Operazione 10

```
(SELECT Cibo.Nome AS Cibo, MAX(Recensione.Stelle) AS Stelle  
FROM Cibo, Recensione  
WHERE Cibo.IDC = Recensione.IDC  
GROUP BY Cibo)
```

UNION

```
(SELECT Bibita.Nome AS Bibita, MAX(Recensione.Stelle) AS Stelle  
FROM Bibita, Recensione  
WHERE Bibita.IDB = Recensione.IDB  
GROUP BY Bibita)
```

ORDER BY Stelle DESC;

--Operazione 11

```
(SELECT IDC AS ID, Nome, Peso, TipoPiatto, IDMenu, Costo, NULL AS Capacità,  
NULL AS Tasso alcolemico  
FROM Cibo)
```

UNION ALL

```
(SELECT IDB AS ID, Nome, NULL AS Peso, NULL AS TipoPiatto, IDMenu,  
Capacità, TassoAlcolemico, Costo  
FROM Bibita)
```

ORDER BY Nome ASC;

--Operazione 12

```
(SELECT Cibo.Nome, COUNT(*) AS "Numero Ordinazioni"  
FROM ComprendeC, Cibo  
WHERE Cibo.IDC = ComprendeC.IDC  
GROUP BY Cibo.Nome)
```

UNION ALL

```
(SELECT Bibita.Nome, COUNT(*) AS "Numero Ordinazioni"  
FROM ComprendeB, Bibita  
WHERE Bibita.IDB = ComprendeB.IDB  
GROUP BY Bibita.Nome)
```

```
ORDER BY "Numero Ordinazioni" DESC  
LIMIT 10;
```

--Operazione 13

```
SELECT I.Nome  
FROM CompostoDa AS CD,Ingredienti AS I  
WHERE I.IDIn = CD.IDC
```

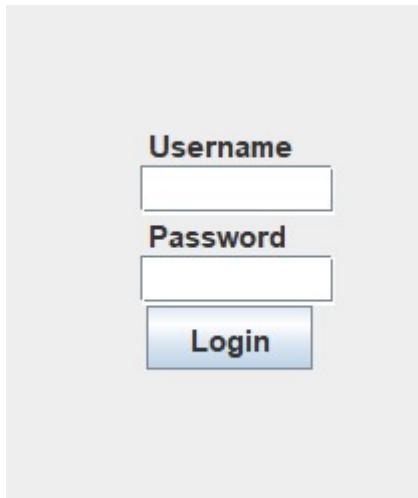
## 6. JDBC

All'avvio dell'applicazione, si simula l'accesso alla piattaforma tramite due account già esistenti, uno è per gestire e simulare i permessi del superuser, mentre un altro per lo

chef, con i permessi descritti nei vincoli delle specifiche della realtà d'interesse. Data l'interattività all'interno del nostro progetto in Java, abbiamo omesso l'esplicito numero dell'operazione effettuata al click del mouse, evidenziando maggiormente l'intelligibilità, l'uso del menu e delle recensioni. L'intenzione principale di questo JDBC è di poter mostrare principalmente il funzionamento della gestione intelligente del ristorante, omettendo volutamente diverse caratteristiche per cercare di dare maggior spazio al fulcro della problematica analizzata.



Sarà qui possibile, nel frame Start, di interagire principalmente sulla scelta del tipo di operazioni da effettuare. Nel “login” abbiamo la possibilità di accedere come chef o come superuser, nel “menu” quella di poter ordinare e visualizzare il menu con tanto di informazioni sugli ingredienti che costituiscono i prodotti, mentre in “visualizza ordini” abbiamo l’insieme degli ordini effettuati ed il conto totale giornaliero, come scritto nella tavola delle operazioni.



Credenziali d'esempio:

username: admin, password: admin

username: chef, password: chef

Pannello dell'Admin

---

Digita IDC:

Digita Nome:

Digita Tipo Piatto:

Digita Costo:

**Inserisci un nuovo Piatto**

---

Digita IDB:

Digita Nome:

Digita Capacità:

Digita Tasso Alcolico:

Digita Costo:

**Inserisci una nuova Bibita**

Pannello dello Chef

Nome: Margherita	Tipo Piatto: Pizza	Costo: 5.0
0	+	-
Nome: Patatine	Tipo Piatto: Frittura	Costo: 2.5
0	+	-
Nome: Tris Crocchè	Tipo Piatto: Frittura	Costo: 5.0
0	+	-
Nome: Diavola	Tipo Piatto: Pizza	Costo: 6.5
0	+	-
Nome: Americana	Tipo Piatto: Pizza	Costo: 7.0
0	+	-
Nome: Tortino	Tipo Piatto: Dolce	Costo: 5.5
0	+	-

**Aggiungi ai Consigliati**

Accedendo alla piattaforma con le credenziali da superuser, si avrà il pannello dell'admin, consentendo all'amministratore la possibilità di creare nuovi piatti o nuove bibite. I permessi sulla visualizzazione del menù con tanto di "Consigliati dallo chef" sono principalmente ed esclusivamente dello chef.

La visualizzazione del menù invece, è molto simile al pannello dello chef: non ha l'aggiunta ai consigliati ma la singola visualizzazione con tanto di ordinazioni effettuabili tramite pulsanti di

incremento e decremento. Vi è anche la possibilità di informarsi sugli ingredienti di un prodotto e di recensirlo.

Le sezioni principali del menù sono quindi piatti preferiti dallo chef, i più recensiti ed il resto dei prodotti, come è scritto nelle specifiche della realtà di interesse.

Questo è un esempio di uno dei tanti prodotti presenti nel menù.

**Categoria: Cibo**

**Nome: Margherita**

**Tipo Piatto: Pizza**

**Costo: 5.0**

**Informati**    **Recensisci**

**Nome: Patatine**

**Scrivi una recensione**

**Ingredienti**

**Sugo**  
**Fior di Latte**  
**Basilico**  
**Parmigiano**

**Invia Recensione**

Infine, per la visualizzazione delle ordinazioni, abbiamo l'ultimo pulsante all'interno del frame Start, ovvero "Visualizza Ordini".

Numero Tavolo: 01  
Data e ora: 2024-01-23 07:07:12  
Conto: 5.0

Conto totale: 5.0