

# UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

**PROGETTO BASI DI DATI A.A 2025/2026**

*TRACCIA 1: SOCIAL*



Acri Stefano N46007605

Allocca Giuseppe N46008040

Di Donna Giuseppe N46007890

<b>1. PRESENTAZIONE.....</b>	<b>4</b>
1.1 Funzionalità del Sistema.....	4
1.2 Obiettivi del progetto.....	5
1.3 Definizione dei dati.....	6
1.4 Definizione delle operazioni.....	8
1.5 Architettura del progetto e dei livelli.....	9
<b>2. PROGETTAZIONE CONCETTUALE.....</b>	<b>10</b>
2.1 Entità e associazioni.....	10
2.2 Cardinalità delle singole associazioni.....	10
2.3 Il modello E/R.....	11
<b>3. PROGETTAZIONE LOGICA.....</b>	<b>12</b>
3.1 Trasformazione.....	12
3.2 Traduzione.....	13
3.3 Modello E/R tradotto.....	14
<b>4. PROGETTAZIONE FISICA.....</b>	<b>16</b>
4.1 Dimensionamento fisico del DB.....	16
4.2 Creazione degli oggetti del DB.....	23
4.3 Vincoli di integrità referenziale.....	25
4.4 Definizione delle politiche di sicurezza e creazione dei ruoli e degli utenti.....	28
4.5 Popolamento della base di dati.....	31
<b>5. ASPETTI FUNZIONALI DEL DB.....</b>	<b>32</b>
5.1 Zona Query e Viste.....	32
5.2 Zona Trigger.....	34
5.3 Zona Stored Procedure.....	37
<b>6. OTTIMIZZAZIONE DEL DB.....</b>	<b>39</b>
6.1 Indici.....	39
6.2 Gestione della concorrenza.....	40
6.3 Gestione dell'affidabilità.....	41
<b>7. INTERFACCIA APEX.....</b>	<b>43</b>
7.1 Integrazione con il Database.....	43
7.2 Accesso e Struttura di Navigazione.....	43
7.3 Home Page: Esplora Attività.....	44
7.4 Scheda Dettaglio e Inserimento Recensioni.....	45
7.5 Gestione Social: Esplora Utenti.....	47
7.6 Il Mio Network: Pagina Amicizie.....	48
7.8 Dashboard Amministrativa e Statistiche.....	49
<b>8. Esecuzione di query in linguaggio naturale.....</b>	<b>51</b>
8.1 Architettura di WrenAI.....	52
8.2 Problemi riscontrati.....	52

# 1.PRESENTAZIONE

Il progetto richiesto consiste nello sviluppo di una piattaforma informatica pensata per la gestione e l'analisi dei dati prodotti da una rete di attività commerciali e dai rispettivi utenti registrati. L'obiettivo è quello di costruire un sistema che replichi le funzionalità di un servizio di recensioni e suggerimenti online, il che impone la necessità di progettare e realizzare una base di dati ben strutturata e coerente.

## 1.1 Funzionalità del Sistema

*Il sistema deve essere in grado di gestire un complesso gruppo di entità e associazioni supportando diversi livelli di accesso e offrendo funzionalità di analisi specifiche:*

- **Gestione delle Entità:** Archiviazione dei dati relativi alle attività commerciali (inclusi nome, ubicazione, coordinate geografiche, valutazione media, stato , orari, categorie di appartenenza e attributi unici) e dei profili utente (con data di registrazione, elenco degli "amici", contatore delle recensioni pubblicate e dei riconoscimenti ricevuti);
- **Gestione delle Interazioni Utente:** Gli utenti hanno la possibilità di pubblicare delle recensioni (specificando data, valutazione, contenuto testuale e "mi piace" ricevuti), pubblicare suggerimenti rapidi, registrare la propria presenza (check-in) presso i locali e caricare immagini (corredate di didascalia ed etichetta);
- **Definizione dei Ruoli:** La piattaforma prevede tre profili di accesso distinti: l'admin (con pieni poteri su tutti i dati), il moderator (autorizzato a esaminare, modificare o eliminare i contenuti generati dagli altri utenti) e lo user (che può creare, modificare o cancellare esclusivamente i propri contenuti);

## 1.2 Obiettivi del progetto

L'obiettivo centrale di questo progetto è creare una piattaforma unificata per la memorizzazione, il monitoraggio e l'analisi delle interazioni tra gli utenti e le diverse attività commerciali.

*Per raggiungere questo scopo, il progetto si articola nelle seguenti fasi chiave:*

- **Progettazione del Database:** Definizione completa dei modelli concettuali, logici e fisici della base di dati.
- **Implementazione SQL e Funzionalità:** Sviluppo in SQL di tutti gli elementi necessari (stored procedure, query, viste e trigger) per supportare le operazioni del sistema, come ad esempio l'aggiornamento automatico della valutazione media di un'attività.
- **Affidabilità e Ottimizzazione del Sistema:** Garanzia di prestazioni e stabilità del sistema attraverso:
  - Definizione di indici per velocizzare le operazioni.
  - Selezione della più adeguata strategia di controllo della concorrenza.
  - Implementazione di meccanismi di backup, recovery e replicazione per l'affidabilità.

## 1.3 Definizione dei dati

La presente documentazione delinea i requisiti per la creazione di una base di dati che possa sostenere una piattaforma di interazione sociale focalizzata sulla valutazione e descrizione di attività commerciali. L'obiettivo è strutturare e organizzare le informazioni provenienti da diverse fonti, incluse le operazioni registrate e i contributi degli utenti.

*Di seguito una lista con le principali richieste della traccia:*

- **Attività Commerciali:** Ogni entità è identificata in modo univoco e richiede la registrazione di dettagli anagrafici (denominazione, indirizzo completo, città, regione, codice postale, coordinate) e indicatori di valutazione(valutazione media, numero di recensioni, stato). È fondamentale categorizzare le attività(es. ristoranti, alberghi) e gestirne i servizi specifici (es. disponibilità Wi-Fi, accessibilità, parcheggio). Devono essere gestiti anche gli orari settimanali delle singole attività.
- **Utenti della Piattaforma:** Gli utenti registrati, ciascuno con un identificativo unico, sono descritti da dati statici (nome, data di iscrizione) e metriche di attività (numero di recensioni pubblicate, numero di complimenti ricevuti, media delle valutazioni sulle proprie recensioni). La piattaforma deve tracciare le connessioni sociali (amicizie con altri utenti) e i vari tipi di apprezzamenti ricevuti alle recensioni scritte(es. "utile", "divertente" o "interessante"). A ogni utente è associato un livello di autorizzazione (utente base, moderatore, amministratore), come descritto nel punto 1.1.

*Gli utenti possono pubblicare diversi tipi di contenuti:*

- **Recensioni:** Opinioni relative a una specifica attività, complete di punteggio numerico, testo esplicativo e data di pubblicazione. Tali valutazioni possono essere a loro volta oggetto di apprezzamento da parte di altri utenti(che le valutano).
- **Suggerimenti Brevi:** Dei suggerimenti associati a un'attività, con data di pubblicazione e conteggio dei complimenti ricevuti.
- **Fotografie:** Immagini univocamente identificate, con didascalia, etichetta(es. "interni", "cibo") e associazione all'utente cariatore e all'attività commerciale di riferimento.

*Il sistema deve registrare:*

- **Check-in:** Registrazione di data e ora delle visite degli utenti presso le entità.
- **Complimenti e Voti:** Memorizzazione dei complimenti e dei voti attribuiti ai contenuti.
- **Relazioni Inter-Utente:** Mappatura della rete sociale interna per modellare le interazioni tra i membri della piattaforma(relazioni di amicizia tra i vari utenti).

## 1.4 Definizione delle operazioni

*Di seguito le principali operazioni previste sul database:*

- Aggiunta di una nuova attività commerciale;
- Revisione delle informazioni di un’attività (categorie, caratteristiche, orari, stato);
- Registrazione di un nuovo utente;
- Aggiornamento degli accessi e privilegi di un utente (utente base, moderatore, amministratore);
- Pubblicazione di una nuova recensione;
- Modifica o cancellazione di una recensione;
- Ricalcolo automatico della valutazione media di un’attività;
- Creazione di un suggerimento rapido;
- Modifica o rimozione di un suggerimento;
- Registrazione di un nuovo check-in presso un’attività;
- Caricamento di una fotografia collegata a un’attività;
- Rimozione di una fotografia;
- Gestione dei legami di amicizia tra utenti;
- Assegnazione di voti o complimenti relativi alle recensioni pubblicate;
- Esecuzione di interrogazioni statistiche sulle attività, sugli utenti o sui check-in;

## 1.5 Architettura del progetto e dei livelli

*L'architettura software del nostro progetto si basa fondamentalmente su una struttura a tre livelli logico-funzionali divisi come segue:*

- **Il livello di presentazione**, è costituito dalle interfacce per gli utenti mediante le quali si possono inserire, visualizzare e gestire i dati relativi alle singole attività commerciali, recensioni, fotografie e check-in. Le interfacce sono principalmente di tipo web a cui vi si può accedere mediante web browser e offrono funzionalità diversificate in funzione del ruolo dell'utente che vi accede;
- **Il livello applicativo**, contiene tutti gli oggetti software responsabili per la logica operativa del sistema. Qui risiedono tutti i componenti che gestiscono la pubblicazione dei contenuti generati dai vari utenti, la gestione delle amicizie tra utenti, calcolo dei valori medi e tutte le operazioni di tipo statistico richieste dal progetto;
- **Il livello dei dati**, è composto dal DBMS , dalla base dati fisica e dalle applicazioni interne che garantiscono la sicurezza, l'integrità e soprattutto la consistenza dei dati. Il DBMS usato è quello di Oracle Database;

*Nel sistema (come descritto parzialmente nei capitoli precedenti), individuiamo 4 categorie di utenti che interagiranno con la base di dati:*

- Il **DBA**(Database Administrator) che dispone di tutti i diritti e privilegi di accesso sulla base di dati, come anche il privilegio per le operazioni di manutenzione, backup e ottimizzazione.
- Gli **admin** che tramite l'opportuna interfaccia possono creare e modificare le singole attività commerciali, aggiornare le informazioni degli utenti e gestire i ruoli(intervenendo sui contenuti del sistema).
- I **moderator**, che attraverso un applicativo web possono intervenire sui contenuti generati dai singoli utenti per approvarli , modificarli o eliminarli (per recensioni , suggerimenti rapidi e fotografie).
- Gli **user**(utenti) effettivi della piattaforma che interagiscono con quest'ultima andando a caricare recensioni/suggerimenti/fotografie per le varie attività commerciali come anche check-in e che possono gestire le proprie amicizie con altri utenti del sistema. L'accesso avviene mediante interfaccia web rispettando i privilegi concessi dal sistema.

## 2. PROGETTAZIONE CONCETTUALE

### 2.1 Entità e associazioni

*Leggendo la traccia del progetto abbiamo individuato le seguenti entità con le rispettive associazioni:*

Si tiene traccia delle **ATTIVITA' COMMERCIALI** che appartengono a una o più **CATEGORIE**.

Ogni **UTENTE** può scrivere una **RECENSIONE** che riguarda un'attività.

**L'UTENTE** può anche caricare una **FOTOGRAFIA** che rappresenta un'attività.

Ogni utente può avere un rapporto di amicizia con gli altri utenti. Ogni attività registra i relativi **CHECK-IN** e offre degli specifici **SERVIZI**(i suoi **ATTRIBUTI**).

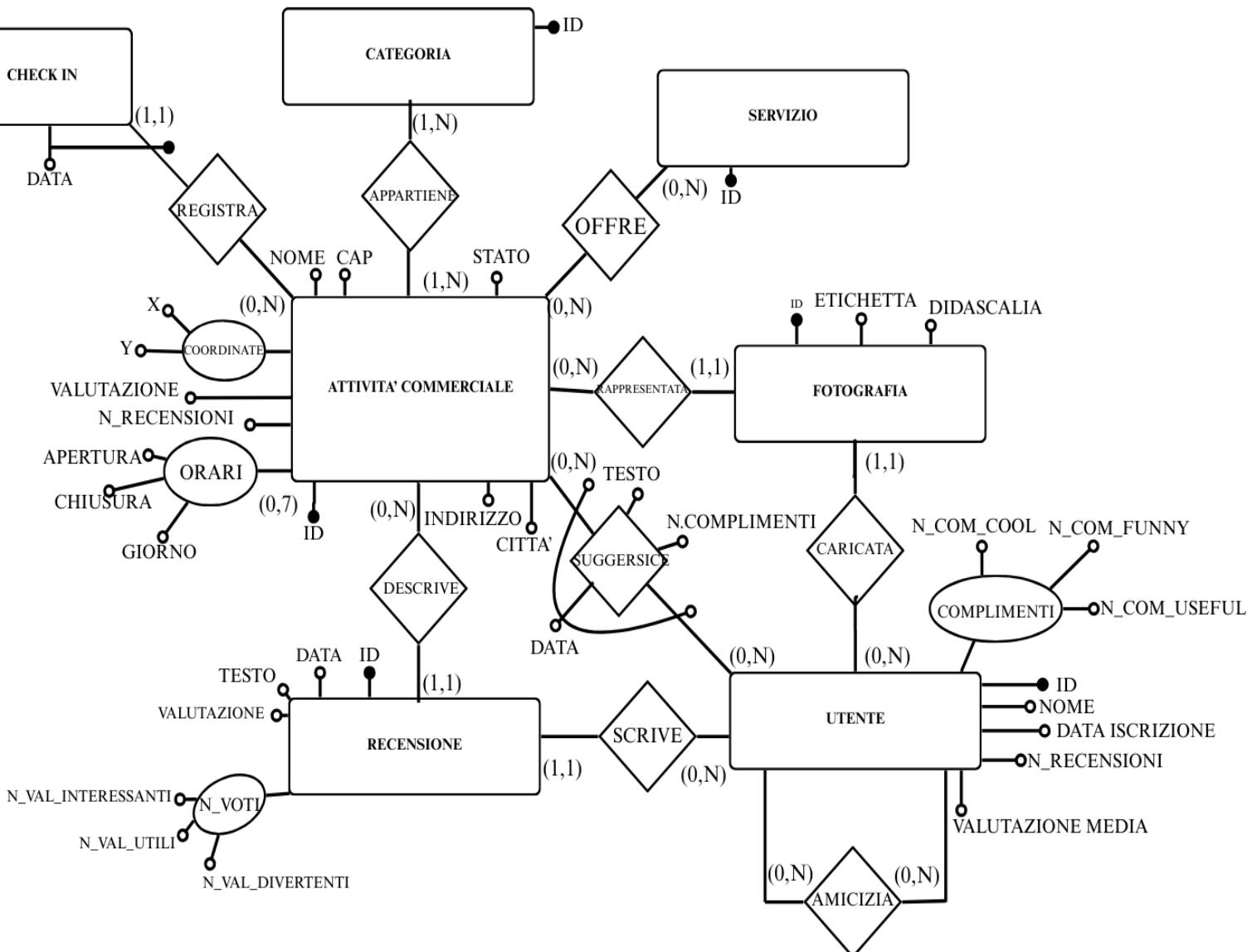
### 2.2 Cardinalità delle singole associazioni

- Un' ATTIVITA' COMMERCIALE può appartenere a una o più CATEGORIE e ogni CATEGORIA può rappresentare una o più ATTIVITA', la cardinalità dell'associazione APPARTIENE è molti a molti;
- Un' ATTIVITA' COMMERCIALE può offrire nessuno o più SERVIZI e ogni SERVIZIO può essere offerto da nessuna o più ATTIVITA' , la cardinalità dell'associazione OFFRE è molti a molti;
- Un' ATTIVITA' COMMERCIALE può registrare nessuno o più CHECK-IN e ogni CHECK-IN può essere registrato da una e una sola ATTIVITA' , la cardinalità dell'associazione REGISTRA è uno a molti;
- Un' ATTIVITA' COMMERCIALE può essere rappresentata da nessuna o più FOTOGRAFIE e ogni FOTOGRAFIA può rappresentare una e una sola ATTIVITA' , la cardinalità dell' associazione RAPPRESENTATA è uno a molti;
- Un' ATTIVITA' COMMERCIALE può essere descritta da nessuna o più RECENSIONI e ogni RECENSIONE può descrivere una e una sola ATTIVITA' , la cardinalità dell'associazione DESCRIVE è uno a molti;
- Un' ATTIVITA' COMMERCIALE può essere suggerita da nessuno o più UTENTI e ogni UTENTE può suggerire nessuna o più ATTIVITA' , la cardinalità dell'associazione SUGGERISCE è molti a molti;

- Un' UTENTE può scrivere nessuna o più RECENSIONI e ogni RECENSIONE può essere scritta da uno e un solo UTENTE , la cardinalità dell'associazione SCRIVE è uno a molti;
- Un' UTENTE può caricare nessuna o più FOTOGRAFIE e ogni FOTOGRAFIA può essere caricata da uno e un solo UTENTE , la cardinalità dell'associazione CARICATA è uno a molti;
- Un' UTENTE può avere nessuno o più UTENTI come amici, la cardinalità dell'associazione AMICIZIA è molti a molti;
- Per l'attributo composto ORARI dell'entità ATTIVITA' abbiamo messo come cardinalità (0,7) per indicare la possibilità di tener traccia dell'orario di chiusura e apertura durante i vari giorni della settimana , più specificatamente il range 0-7 è dato dalla possibilità di un'attività commerciale di esercitare il proprio servizio da 0 a 7 giorni;

## 2.3 Il modello E/R

Ecco di seguito il modello E/R che abbiamo realizzato seguendo la traccia:



### 3. PROGETTAZIONE LOGICA

La progettazione logica ha l'obiettivo di definire l'architettura tabellare del database (schema relazionale). Il workflow prevede due passaggi sequenziali:

- Step 1: Trasformazione. Si "pulisce" il modello concettuale (ER) eliminando elementi incompatibili con le tabelle, quali attributi composti e/o multivale;re;
- Step 2: Traduzione. Si mappano le entità e le associazioni rimanenti in vere e proprie relazioni (tabelle), definendo chiavi e vincoli;

#### 3.1 Trasformazione

Applicando le regole di trasformazione al nostro progetto, questa fase si concentra su due interventi correttivi: **l'eliminazione degli attributi composti e la trasformazione degli attributi multivale.**

##### 1. *Eliminazione degli Attributi Composti:*

In questa fase, gli attributi composti vengono rimossi e sostituiti direttamente dai sotto-attributi che li compongono :

- **Entità RECENSIONE:** L'attributo composto *N\_Voti* scompare; al suo posto vengono inseriti esplicitamente i sotto-attributi: *N\_Val\_Utili*, *N\_Val\_Divertenti* e *N\_Val\_Interessanti*.
- **Entità UTENTE:** Analogamente, l'attributo *Complimenti* viene scomposto nei suoi sotto-attributi: *Num\_Complimenti\_Cool*, *Num\_Complimenti\_Funny* e *Num\_Complimenti\_Usedful*.
- **Entità ATTIVITÀ:** Le coordinate geografiche non sono più trattate come un attributo unico (*Coordinate*), ma scisse in due attributi distinti: *Latitudine* e *Longitudine*(espressi nel modello come X e Y).

## 2. Gestione dell'Attributo Multivalore (*Orari*):

L'attributo *Orari*, che definisce gli orari di apertura, presenta una cardinalità multipla (un'attività è aperta più giorni alla settimana) che il modello relazionale non supporta in un singolo campo.

- **Soluzione:** Si crea una nuova associazione(*Rispetta*) che collega l'entità che originariamente possedeva l'attributo multivalore e la nuova entità che rappresenta il precedente ed eliminato attributo(*Orari*);

## 3.2 Traduzione

Questa fase converte lo schema ristrutturato in uno schema logico definitivo. Le regole applicate sono le seguenti:

### 1. Traduzione delle Entità

Ogni entità viene convertita in una relazione (tabella) indipendente.

- **Nome:** Acquisisce il nome dell'entità originale, declinato al plurale.
- **Colonne:** Includono tutti gli attributi semplici dell'entità.
- **Chiave Primaria (PK):** Corrisponde all' attributo identificatore univoco dell'entità di partenza.

### 2. Traduzione delle Associazioni Molti-a-Molti (N:N)

Le associazioni con cardinalità molti-a-molti generano nuove tabelle autonome.

- **Struttura:** La nuova relazione contiene gli attributi propri dell'associazione e gli identificatori delle entità coinvolte.
- **Chiave Primaria:** È composta dalla composizione delle chiavi primarie delle entità associate e ovviamente ci sono i vincoli referenziali.
- **Applicazione al progetto:** Le associazioni APPARTIENE, OFFRE, SUGGERISCE E AMICIZIA diventano delle relazioni a sé stanti.

### 3. Traduzione delle Associazioni Uno-a-Molti (1:N)

Queste associazioni vengono risolte cosicché si aggiunga alla relazione che identifica l'entità dal lato uno , l'identificatore dell'entità al lato molti(con vincolo referenziale) più tutti gli attributi dell'associazione.

*Applicazione al progetto:*

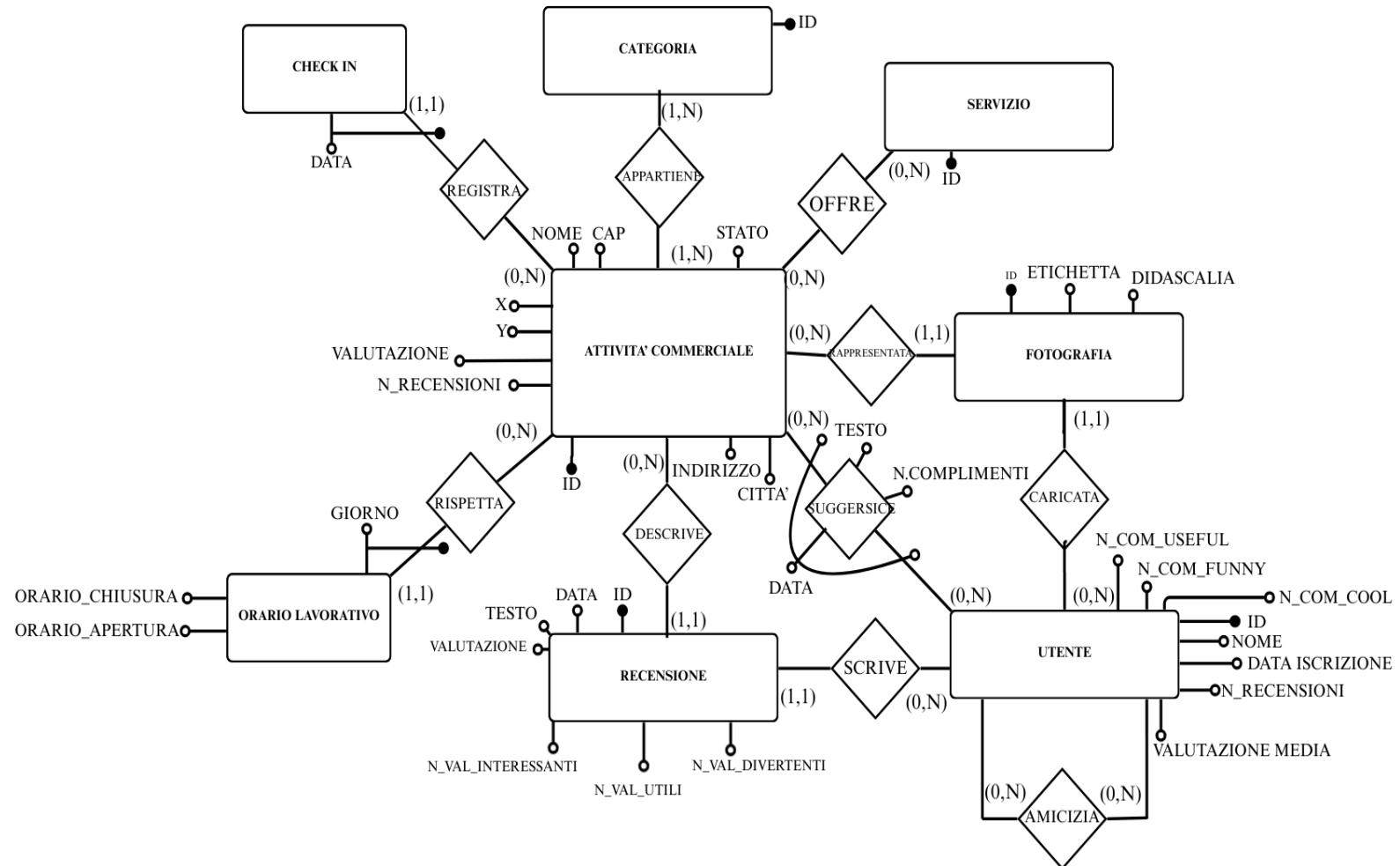
- **Le associazioni CARICATA e RAPPRESENTATA spariscono:** gli identificatori di UTENTE e ATTIVITA' COMMERCIALE sono aggiunti alla relazione FOTOGRAFIA.
- **Le associazioni DESCRIVE e SCRIVE spariscono:** gli identificatori di ATTIVITA' COMMERCIALE e di UTENTE sono aggiunti alla relazione RECENSIONE.
- **L'associazione REGISTRA sparisce:** l'identificatore di ATTIVITA' COMMERCIALE viene aggiunto alla relazione CHECK IN.

### 3.3 Modello E/R tradotto

*Di seguito è riportato lo schema relazionale completo per la base di dati in oggetto:*

- **ATTIVITA'\_COMMERCIALI** (ID, INDIRIZZO, CITTA', NOME, CAP, STATO, X ,Y, VALUTAZIONE\_MEDIA , N\_RECENSIONI);
- **RECENSIONI** (ID, DATA, TESTO, VALUTAZIONE, N\_VAL\_INTERESSANTI, N\_VAL\_UTILI, N\_VAL\_DIVERTENTI, ATTIVITA' : ATTIVITA'\_COMMERCIALI, UTENTE : UTENTI);
- **UTENTI** (ID, NOME , DATA\_ISCRIZIONE, NUM\_RECENSIONI, NUM\_COMPLIMENTI\_COOL ,NUM\_COMPLIMENTI\_FUNNY , NUM\_COMPLIMENTI\_USEFUL, VALUTAZIONE\_MEDIA);
- **FOTOGRAFIE** (ID, ETICHETTA, TESTO, IMMAGINE , ATTIVITA' : ATTIVITA' COMMERCIALI, UTENTE : UTENTI);
- **SERVIZI** (ID);
- **CATEGORIE** (ID);
- **CHECK\_IN** (DATA, ATTIVITA':ATTIVITA' COMMERCIALI);
- **ORARI LAVORATIVI** (GIORNO , ATTIVITA':  
ATTIVITA'\_COMMERCIALI , ORARIO\_APERTURA , ORARIO\_CHIUSURA);
- **SUGGERIMENTI** (DATA , UTENTE : UTENTE, ATTIVITA' :  
ATTIVITA'\_COMMERCIALI, TESTO , NUM\_COMPLIMENTI);

- **APPARTENENZE\_CATEGORIE** (ATTIVITA' :  
ATTIVITA'\_COMMERCIALI, CATEGORIA : CATEGORIE);
- **OFFERTE\_SERVIZI** (SERVIZIO : SERVIZI , ATTIVITA' :  
ATTIVITA'\_COMMERCIALI);
- **AMICIZIE** (UTENTE : UTENTI , AMICO : UTENTI);



## 4. PROGETTAZIONE FISICA

### 4.1 Dimensionamento fisico del DB

A valle della definizione dei volumi di dati previsti e della scelta dei tipi di attributo, è necessario effettuare una stima preventiva dell'occupazione di memoria, passaggio fondamentale per la successiva implementazione fisica del database.

Tale analisi garantisce un dimensionamento corretto delle risorse di storage.

*Facendo riferimento al DBMS Oracle, le metriche di occupazione per i tipi di dati utilizzati sono le seguenti:*

- **VARCHAR2(x)**: x byte.
- **DATE**: 7 byte fissi.
- **TIMESTAMP**: 11 byte fissi.
- **LOB (CLOB/BLOB)**: 86 byte (dimensione del locator).
- **NUMBER/INTEGER(x)**:  $(x/2+2)$  byte.

Il calcolo dell'occupazione totale per ciascuna tabella segue una procedura a due step:

1. Calcolo dell'occupazione per riga : si sommano i byte richiesti da ogni attributo della tabella utilizzando le regole specifiche al tipo di dato sopra riportate;
2. Stima dell'occupazione totale: Si moltiplica la dimensione del singolo record per il numero stimato di tuple (righe) della tabella.

## **1. ATTIVITA\_COMMERCIALI**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
ID_Attivita	VARCHAR2(30)	30
Nome	VARCHAR2(100)	100
Valutazione_Media	NUMBER	3
Indirizzo	VARCHAR2(150)	150
Città	VARCHAR2(50)	50
CAP	VARCHAR2(50)	50
Stato	VARCHAR2(10)	10
Latitudine	NUMBER	13
Longitudine	NUMBER	13
Num_Recensioni	INTEGER	7
<b>TOTALE PER RIGA</b>		<b>426</b>

## **2. CATEGORIE**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
ID_Categoria	VARCHAR2(50)	50
<b>TOTALE PER RIGA</b>		<b>50</b>

### **3. APPARTENENZE\_CATEGORIE**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
Attivita	VARCHAR2(30)	30
Categoria	VARCHAR2(50)	50
<b>TOTALE PER RIGA</b>		<b>80</b>

### **4. OFFERTE\_SERVIZI**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
Attivita	VARCHAR2(30)	30
Servizio	VARCHAR2(50)	50
<b>TOTALE PER RIGA</b>		<b>80</b>

### **5. ORARI\_LAVORATIVI**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
Attivita	VARCHAR2(30)	30
Giorno	VARCHAR2(10)	10
Apertura	VARCHAR2(10)	10
Chiusura	VARCHAR2(10)	10
<b>TOTALE PER RIGA</b>		<b>60</b>

## 6. FOTOGRAFIE

Attributo	Tipo	Byte (Max)
ID_Fotografia	VARCHAR2(30)	30
Didascalia	VARCHAR2(250)	250
Etichetta	VARCHAR2(50)	50
Contenuto	BLOB	86
Attivita	VARCHAR2(30)	30
Utente	VARCHAR2(30)	30
<b>TOTALE PER RIGA</b>		<b>476</b>

## 7. UTENTI

Attributo	Tipo	Byte (Max)
ID_Utente	VARCHAR2(30)	30
Nome_Utente	VARCHAR2(50)	50
Data_Iscrizione	DATE	7
Num_Recensioni	INTEGER	7
Num_Complimenti_Cool	INTEGER	7
Num_Complimenti_Funny	INTEGER	7
Num_Complimenti_Useful	INTEGER	7
Valutazione_Media	NUMBER	3
<b>TOTALE PER RIGA</b>		<b>118</b>

## **8. AMICIZIE**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
Utente	VARCHAR2(30)	30
Amico	VARCHAR2(30)	30
<b>TOTALE PER RIGA</b>		<b>60</b>

## **9. RECENSIONI**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
ID_Recensione	VARCHAR2(30)	30
Valutazione	NUMBER	3
Data	DATE	7
Testo	CLOB	86
Num_Valutazioni_Utili	INTEGER	7
Num_Valutazioni_Divertenti	INTEGER	7
Num_Valutazioni_Interessanti	INTEGER	7
Attivita	VARCHAR2(30)	30
Utente	VARCHAR2(30)	30
<b>TOTALE PER RIGA</b>		<b>207</b>

## **10. SUGGERIMENTI**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
Testo	VARCHAR2(500)	500
Data	DATE	7
Num_Complimenti	INTEGER	7
Attivita	VARCHAR2(30)	30
Utente	VARCHAR2(30)	30
<b>TOTALE PER RIGA</b>		<b>574</b>

## **11. CHECK\_IN**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
Data	TIMESTAMP	11
Attivita	VARCHAR2(30)	30
<b>TOTALE PER RIGA</b>		<b>41</b>

## **12. SERVIZI**

<b>Attributo</b>	<b>Tipo</b>	<b>Byte (Max)</b>
ID_Servizio	VARCHAR2(50)	50
<b>TOTALE PER RIGA</b>		<b>50</b>

### **13. DIMENSIONAMENTO TOTALE NEL WORST CASE:**

Tabella	Occupazione Totale (Byte)
<b>ATTIVITA_COMMERCIALI</b>	$426 * N_{Righe}$
<b>CATEGORIE</b>	$50 * N_{Righe}$
<b>SERVIZI</b>	$50 * N_{Righe}$
<b>FOTOGRAFIE</b>	$476 * N_{Righe}$
<b>UTENTI</b>	$118 * N_{Righe}$
<b>RECENSIONI</b>	$207 * N_{Righe}$
<b>CHECK_IN</b>	$41 * N_{Righe}$
<b>ORARI_LAVORATIVI</b>	$60 * N_{Righe}$
<b>SUGGERIMENTI</b>	$574 * N_{Righe}$
<b>APPARTENENZE_CATEGORIE</b>	$80 * N_{Righe}$
<b>OFFERTE_SERVIZI</b>	$80 * N_{Righe}$
<b>AMICIZIE</b>	$60 * N_{Righe}$
<b>TOTALE COMPLESSIVO</b>	$1796 * N_{Righe}$

## 4.2 Creazione degli oggetti del DB

```
-- CREAZIONE DELLE RELAZIONI --
-- Creazione Tabella ATTIVITA_COMMERCIALI
CREATE TABLE ATTIVITA_COMMERCIALI (
    ID_Attivita VARCHAR2(30),
    Nome VARCHAR2(100) NOT NULL,
    Valutazione_Media NUMBER DEFAULT 0,
    Indirizzo VARCHAR2(150),
    Città VARCHAR2(50)NOT NULL,
    CAP VARCHAR2(50),
    Stato VARCHAR2(10),
    Latitudine NUMBER,
    Longitudine NUMBER,
    Num_Recensioni INTEGER DEFAULT 0,
    CONSTRAINT PK_ID_Attivita PRIMARY KEY (ID_Attivita),
    CONSTRAINT Check_Valutazione_Media CHECK (Valutazione_Media >= 0 AND Valutazione_Media <= 5)
);

-- Creazione Tabella CATEGORIE
CREATE TABLE CATEGORIE (
    ID_Categoria VARCHAR2(50),
    CONSTRAINT PK_ID_Categoria PRIMARY KEY (ID_Categoria)
);

-- Creazione Tabella SERVIZI
CREATE TABLE SERVIZI (
    ID_Servizio VARCHAR2(50),
    CONSTRAINT PK_ID_Servizio PRIMARY KEY (ID_Servizio)
);

-- Creazione Tabella APPARTENENZE_CATEGORIE
CREATE TABLE APPARTENENZE_CATEGORIE(
    Attivita VARCHAR2(30),
    Categoria VARCHAR2(50),
    CONSTRAINT PK_Appartenza_Categoria PRIMARY KEY(Attivita, Categoria)
);

-- Creazione Tabella OFFERTE_SERVIZI
CREATE TABLE OFFERTE_SERVIZI(
    Servizio VARCHAR2(30),
    Attivita VARCHAR2(50),
    CONSTRAINT PK_Offerta_Servizio PRIMARY KEY(Servizio, Attivita)
);

-- Creazione Tabella CHECK_IN
CREATE TABLE CHECK_IN(
    Data TIMESTAMP,
    Attivita VARCHAR2(30),
    CONSTRAINT PK_Check_In PRIMARY KEY (Data, Attivita)
);

-- Creazione Tabella ORARI_LAVORATIVI
CREATE TABLE ORARI_LAVORATIVI(
    Giorno VARCHAR2(30),
    Attivita VARCHAR2(30),
    Orario_Apertura VARCHAR2(5),
    Orario_Chiusura VARCHAR2(5),
    CONSTRAINT PK_Orario_Lavorativo PRIMARY KEY (Giorno, Attivita)
);
```

```

-- Creazione Tabella FOTOGRAFIE
CREATE TABLE FOTOGRAFIE(
ID_Fotografia VARCHAR2(30),
Testo CLOB,
Immagine BLOB,
Etichetta VARCHAR2(50),
Attivita VARCHAR2(30) NOT NULL,
Utente VARCHAR2(30), --Puo' essere NULL
CONSTRAINT PK_ID_Fotografia PRIMARY KEY (ID_Fotografia)
);

-- Creazione Tabella UTENTI
CREATE TABLE UTENTI(
ID_Utente VARCHAR2(30),
Nome VARCHAR2(50) NOT NULL,
Data_Iscrizione DATE DEFAULT SYSDATE,
Num_Recensioni INTEGER DEFAULT 0,
Num_Complimenti_Cool INTEGER DEFAULT 0,
Num_Complimenti_Funny INTEGER DEFAULT 0,
Num_Complimenti_Usedful INTEGER DEFAULT 0,
Valutazione_Media_Recensioni NUMBER DEFAULT 0,
CONSTRAINT Check_Valutazione_Media_Recensioni CHECK (Valutazione_Media_Recensioni >= 0 AND Valutazione_Media_Recensioni <= 5),
CONSTRAINT PK_ID_Utente PRIMARY KEY (ID_Utente)
);

-- Creazione Tabella AMICIZIE
CREATE TABLE AMICIZIE(
Utente VARCHAR2(30),
Amico VARCHAR2(30),
CONSTRAINT PK_Amicizia PRIMARY KEY(Utente, Amico)
);

-- Creazione Tabella RECENSIONI
CREATE TABLE RECENSIONI(
ID_Recensione VARCHAR2(30),
Data TIMESTAMP DEFAULT SYSDATE,
Testo CLOB,
Valutazione INTEGER NOT NULL,
Num_Valutazioni_Divertenti INTEGER DEFAULT 0,
Num_Valutazioni_Interessanti INTEGER DEFAULT 0,
Num_Valutazioni_Utili INTEGER DEFAULT 0,
Attivita VARCHAR2(30) NOT NULL,
Utente VARCHAR2(30) NOT NULL,
CONSTRAINT PK_ID_Recensione PRIMARY KEY (ID_Recensione),
CONSTRAINT Check_Valutazione CHECK (Valutazione >= 0 AND Valutazione <= 5)
);

-- Creazione Tabella SUGGERIMENTI
CREATE TABLE SUGGERIMENTI(
Data DATE,
Utente VARCHAR2(30),
Attivita VARCHAR2(30),
Testo CLOB NOT NULL,
Num_Complimenti INTEGER DEFAULT 0,
CONSTRAINT PK_Suggerimento PRIMARY KEY (Data, Utente, Attivita)
);

```

## 4.3 Vincoli di integrità referenziale

Al fine di preservare la solidità strutturale del database, sono stati applicati i vincoli di integrità referenziale.

Tale meccanismo assicura la sincronia tra i record, subordinando la validità di una Foreign Key all'effettiva preesistenza della corrispettiva Primary Key (fatti salvi i casi di opzionalità tramite valori NULL).

La scelta metodologica è stata quella di iniettare tali logiche di controllo in una fase successiva alla creazione stessa delle tabelle, operando una riconfigurazione dello schema tramite l'istruzione ALTER TABLE come visualizzabile dagli screen sottostanti:

```
-- VINCOLI D'INTEGRITA' REFERENZIALE --
-- Definizione delle Foreign Key per la tabella FOTOGRAFIE
ALTER TABLE FOTOGRAFIE ADD CONSTRAINT FK_FOTOGRAFIE_ATTIVITA
FOREIGN KEY(Attivita) REFERENCES ATTIVITA_COMMERCIALI(ID_Attivita)
ON DELETE CASCADE;

ALTER TABLE FOTOGRAFIE ADD CONSTRAINT FK_FOTOGRAFIE_UTENTI
FOREIGN KEY(Utente) REFERENCES UTENTI(ID_Utente)
ON DELETE CASCADE;

-- Definizione delle Foreign Key per la tabella RECENSIONI
ALTER TABLE RECENSIONI ADD CONSTRAINT FK_RECENSIONI_ATTIVITA
FOREIGN KEY(Attivita) REFERENCES ATTIVITA_COMMERCIALI(ID_Attivita)
ON DELETE CASCADE;

ALTER TABLE RECENSIONI ADD CONSTRAINT FK_RECENSIONI_UTENTI
FOREIGN KEY(Utente) REFERENCES UTENTI(ID_Utente)
ON DELETE CASCADE;

-- Definizione delle Foreign Key per la tabella CHECK_IN
ALTER TABLE CHECK_IN ADD CONSTRAINT FK_CHECKIN_ATTIVITA
FOREIGN KEY(Attivita) REFERENCES ATTIVITA_COMMERCIALI(ID_Attivita)
ON DELETE CASCADE;

-- Definizione delle Foreign Key per la tabella ORARI_LAVORATIVI
ALTER TABLE ORARI_LAVORATIVI ADD CONSTRAINT FK_ORARI_ATTIVITA
FOREIGN KEY(Attivita) REFERENCES ATTIVITA_COMMERCIALI(ID_Attivita)
ON DELETE CASCADE;
```

```

-- Definizione delle Foreign Key per la tabella SUGGERIMENTI
ALTER TABLE SUGGERIMENTI ADD CONSTRAINT FK_SUGGERIMENTI_ATTIVITA
FOREIGN KEY(Attivita) REFERENCES ATTIVITA_COMMERCIALI(ID_Attivita)
ON DELETE CASCADE;

ALTER TABLE SUGGERIMENTI ADD CONSTRAINT FK_SUGGERIMENTI_UTENTI
FOREIGN KEY(Utente) REFERENCES UTENTI(ID_Utente)
ON DELETE CASCADE;

-- Definizione delle Foreign Key per la tabella APPARTENENZE_CATEGORIE
ALTER TABLE APPARTENENZE_CATEGORIE ADD CONSTRAINT FK_APP_ATTIVITA
FOREIGN KEY(Attivita) REFERENCES ATTIVITA_COMMERCIALI(ID_Attivita)
ON DELETE CASCADE;

ALTER TABLE APPARTENENZE_CATEGORIE ADD CONSTRAINT FK_APP_CATEGORIE
FOREIGN KEY(Categoria) REFERENCES CATEGORIE(ID_Categoria)
ON DELETE CASCADE;

-- Definizione delle Foreign Key per la tabella OFFERTE_SERVIZI
ALTER TABLE OFFERTE_SERVIZI ADD CONSTRAINT FK_OFFERTE_ATTIVITA
FOREIGN KEY(Attivita) REFERENCES ATTIVITA_COMMERCIALI(ID_Attivita)
ON DELETE CASCADE;

ALTER TABLE OFFERTE_SERVIZI ADD CONSTRAINT FK_OFFERTE_SERVIZI
FOREIGN KEY(Servizio) REFERENCES SERVIZI(ID_Servizio)
ON DELETE CASCADE;

-- Definizione delle Foreign Key per la tabella AMICIZIE
ALTER TABLE AMICIZIE ADD CONSTRAINT FK_AMICIZIE_UTENTI
FOREIGN KEY(Utente) REFERENCES UTENTI(ID_Utente)
ON DELETE CASCADE;

```

*L'utilizzo dei vincoli di chiave esterna(FK) è stata guidata dal fatto che fosse necessario mantenere una certa consistenza nel DB. Di conseguenza , illustriamo qui di seguito le logiche adottate per le varie entità:*

## **Eliminazione a cascata (ON DELETE CASCADE):**

Per la maggior parte delle relazioni si è usata la clausola ON DELETE CASCADE.

Questo è dovuto dal fatto che ci sono delle dipendenze forti tra le entità: se l'entità referenziata è rimossa di conseguenza anche le entità referenzianti perdono il loro significato e devono essere rimosse di conseguenza per evitare dati “orfani”.

*Nello specifico, analizziamo le seguenti dipendenze riflettute nelle associazioni tra entità:*

- **Entità Strettamente Dipendenti (Orari lavorativi, Check-in, Offerte Servizi):**  
Le tabelle ORARI\_LAVORATIVI, CHECK\_IN e OFFERTE SERVIZI sono logicamente subordinate all'esistenza di un'attività commerciale. Qualora un'impresa venga rimossa dal sistema, i relativi orari, lo storico degli accessi e l'elenco delle prestazioni offerte vengono eliminati automaticamente, evitando la persistenza di record orfani e privi di contesto.
- **Tabelle di Relazione (Appartenenze Categorie):**  
L'entità APPARTENENZE\_CATEGORIE gestisce l'associazione molti-a-molti tra le attività e la categoria relativa.  
In questa configurazione, la cancellazione di una categoria o di una specifica attività comporta lo scioglimento immediato e automatico del legame tra le due entità, mantenendo lo schema relazionale pulito e aggiornato.
- **Contenuti Generati dagli Utenti (Recensioni, Suggerimenti, Fotografie):**  
Il sistema garantisce la coerenza tra i contributi e le anagrafiche. In caso di chiusura definitiva di un'attività, tutte le recensioni , le fotografie e i suggerimenti ad essa collegate vengono rimosse.  
Allo stesso modo, per garantire la protezione dei dati e la coerenza del database, l'eliminazione del profilo di un utente comporta la rimozione contestuale di ogni suo contributo (suggerimenti , recensioni e fotografie).

## Gestione delle AMICIZIE

*Nella tabella AMICIZIE, la gestione dei vincoli è stata adattata come segue:*

- È presente il vincolo con ON DELETE CASCADE. Se un utente viene rimosso, tutte le sue relazioni di amicizia vengono cancellate.

## 4.4 Definizione delle politiche di sicurezza e creazione dei ruoli e degli utenti

*All'interno del database sono stati definiti i seguenti ruoli:*

- **RUOLO\_ADMIN:** Il ruolo di amministratore della base di dati è fondamentale affinché in futuro quest'ultima possa essere soggetta a processi di aggiornamento, revisione e manutenzione.  
Da un punto di vista pratico, l'amministratore è colui a cui viene garantito l'accesso in lettura e modifica (SELECT, INSERT, UPDATE e DELETE) su tutte le principali tabelle del sistema, infatti se così non fosse lo sviluppo e l'amministrazione generale della base di dati non potrebbero essere garantite.
- **RUOLO\_MODERATOR:** Il ruolo di moderatore è stato progettato con l'intento di aiutare la piattaforma a rimanere il più “pulita” possibile da eventuali contenuti inappropriati degli utenti. Per tale motivo il moderatore ha il privilegio di lettura su tutte le tabelle principali del sistema, mentre i privilegi di modifica o cancellazione gli sono stati concessi soltanto per adempiere al suo compito vale a dire quello di supervisore di RECENSIONI, FOTOGRAFIE e SUGGERIMENTI.
- **RUOLO\_USER:** Il ruolo utente è quello predisposto a rappresentare la persona che dovrà semplicemente accedere alla piattaforma e usufruire dei servizi da essa proposti, pertanto i privilegi garantiti per un utente riguardano strettamente quelle tabelle utili a descrivere l'interazione con il sistema ed altri utenti di pari livello

Per la realizzazione di test di vario tipo sono stati creati degli utenti dedicati (ad esempio Admin\_della\_Piattaforma oppure Moderatore\_della\_Piattaforma), uno per categoria, in modo da simulare correttamente i tre profili descritti precedentemente.

Tale necessità si è presentata anche durante la progettazione della GUI.

### **Creazione dei ruoli nel DB:**

```
-- CREAZIONE DEI RUOLI E DEGLI UTENTI --
-- Definizione del ruolo ADMIN (Gestione completa del sistema)
CREATE ROLE RUOLO_ADMIN;

-- Definizione del ruolo MODERATOR (Gestione contenuti degli utenti)
CREATE ROLE RUOLO_MODERATOR;

-- Definizione del ruolo USER (Gestione dei propri)
CREATE ROLE RUOLO_USER;
```

### **Cessione dei privilegi al ruolo di admin , moderator e user:**

```
-- L'ADMIN può gestire in maniera completa tutto il sistema.
-- Concediamo a tale ruolo privilegi completi su tutte le tabelle
GRANT ALL PRIVILEGES ON ATTIVITA_COMMERCIALI TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON CATEGORIE TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON SERVIZI TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON FOTOGRAFIE TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON UTENTI TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON RECENSIONI TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON CHECK_IN TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON ORARI_LAVORATIVI TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON SUGGERIMENTI TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON APPARTENENZE_CATEGORIE TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON OFFERTE_SERVIZI TO RUOLO_ADMIN;
GRANT ALL PRIVILEGES ON AMICIZIE TO RUOLO_ADMIN;

-- Il MODERATOR può gestire soltanto i contenuti di altri utenti.
-- Concediamo a tale ruolo il privilegio di SELEZIONE su tutte le tabelle della base di dati
GRANT SELECT ON ATTIVITA_COMMERCIALI TO RUOLO_MODERATOR;
GRANT SELECT ON CATEGORIE TO RUOLO_MODERATOR;
GRANT SELECT ON SERVIZI TO RUOLO_MODERATOR;
GRANT SELECT ON FOTOGRAFIE TO RUOLO_MODERATOR;
GRANT SELECT ON UTENTI TO RUOLO_MODERATOR;
GRANT SELECT ON RECENSIONI TO RUOLO_MODERATOR;
GRANT SELECT ON CHECK_IN TO RUOLO_MODERATOR;
GRANT SELECT ON ORARI_LAVORATIVI TO RUOLO_MODERATOR;
GRANT SELECT ON SUGGERIMENTI TO RUOLO_MODERATOR;
GRANT SELECT ON APPARTENENZE_CATEGORIE TO RUOLO_MODERATOR;
GRANT SELECT ON OFFERTE_SERVIZI TO RUOLO_MODERATOR;
GRANT SELECT ON AMICIZIE TO RUOLO_MODERATOR;

-- Concediamo a tale ruolo il privilegio di CANCELLAZIONE e MODIFICA su alcune tabelle della base di dati
GRANT DELETE ON FOTOGRAFIE TO RUOLO_MODERATOR;
GRANT DELETE ON SUGGERIMENTI TO RUOLO_MODERATOR;
GRANT DELETE ON CHECK_IN TO RUOLO_MODERATOR;
GRANT DELETE ON RECENSIONI TO RUOLO_MODERATOR;
GRANT UPDATE ON FOTOGRAFIE TO RUOLO_MODERATOR;
GRANT UPDATE ON SUGGERIMENTI TO RUOLO_MODERATOR;
GRANT UPDATE ON CHECK_IN TO RUOLO_MODERATOR;
GRANT UPDATE ON RECENSIONI TO RUOLO_MODERATOR;
```

```

-- Lo USER può gestire soltanto i propri contenuti.
-- Concediamo a tale ruolo il privilegio di SELEZIONE su alcune tabelle della base di dati
GRANT SELECT ON ATTIVITA_COMMERCIALI TO RUOLO_USER;
GRANT SELECT ON CATEGORIE TO RUOLO_USER;
GRANT SELECT ON SERVIZI TO RUOLO_USER;
GRANT SELECT ON FOTOGRAFIE TO RUOLO_USER;
GRANT SELECT ON UTENTI TO RUOLO_USER;
GRANT SELECT ON RECENSIONI TO RUOLO_USER;
GRANT SELECT ON CHECK_IN TO RUOLO_USER;
GRANT SELECT ON ORARI_LAVORATIVI TO RUOLO_USER;
GRANT SELECT ON SUGGERIMENTI TO RUOLO_USER;
GRANT SELECT ON AMICIZIE TO RUOLO_USER;

-- Concediamo a tale ruolo il privilegio di INSERIMENTO ed AGGIORNAMENTO su alcune tabelle della base di dati
GRANT INSERT ON FOTOGRAFIE TO RUOLO_USER;
GRANT INSERT ON SUGGERIMENTI TO RUOLO_USER;
GRANT INSERT ON RECENSIONI TO RUOLO_USER;
GRANT INSERT ON CHECK_IN TO RUOLO_USER;
GRANT INSERT ON AMICIZIE TO RUOLO_USER;
GRANT UPDATE ON SUGGERIMENTI TO RUOLO_USER;
GRANT UPDATE ON RECENSIONI TO RUOLO_USER;
GRANT UPDATE ON FOTOGRAFIE TO RUOLO_USER;

```

### ***Creazione di un admin , di un moderatore e di un utente:***

-- Creazione di un utente ADMIN

```

CREATE USER Admin_della_Piattaforma IDENTIFIED BY adm;
GRANT CREATE SESSION TO Admin_della_Piattaforma;
GRANT RUOLO_ADMIN TO Admin_della_Piattaforma;

```

-- Creazione di un utente MODERATOR

```

CREATE USER Moderatore_della_Piattaforma IDENTIFIED BY modrt;
GRANT CREATE SESSION TO Moderatore_della_Piattaforma;
GRANT RUOLO_MODERATOR TO Moderatore_della_Piattaforma;

```

-- Creazione di un utente USER

```

CREATE USER User_della_Piattaforma IDENTIFIED BY usr;
GRANT CREATE SESSION TO User_della_Piattaforma;
GRANT RUOLO_USER TO User_della_Piattaforma ;

```

Pertanto tramite Oracle APEX, si è estesa la gestione della sicurezza all'interfaccia web mediante degli appositi strumenti:

- **Application Access Control:** Il riconoscimento dei privilegi da garantire agli utenti è completamente delegato agli account APEX, organizzati in gruppi, che coincidono esattamente con i ruoli illustrati precedentemente
- **Authorization Schemes:** Sono stati creati schemi di autorizzazione dinamici che condizionano la visibilità degli elementi dell'interfaccia. Ad esempio la tab denominata “Dashboard Amministrativa” in cui sono presenti dei grafici statistici e delle classifiche Top 10 riguardanti i dati dell'intero sistema, sono visibili soltanto agli utenti con account admin

## 4.5 Popolamento della base di dati

Una volta terminata la progettazione fisica della base di dati, tramite le creazione delle tabelle già presentate, bisognava popolare l'infrastruttura realizzata.

Il processo di popolamento è stato possibile grazie al dataset Yelp fornитoci, purtroppo però a causa di esigenze operative e limitazioni hardware non è stato possibile utilizzare l'intera raccolta dati ma soltanto un piccola parte di essa.

### Conversione del formato

Il dataset iniziale era strutturato in più file JSON che grazie al materiale didattico (script conversion\_JSON\_CSV.py) abbiamo potuto “tradurre” in degli omonimi file CSV che garantivano una migliore organizzazione dei dati in vista della futura importazione su Oracle SQL Developer.

Questa prima fase è stata fondamentale per avere a disposizione dei file improntati ad una struttura tabellare molto simile a quella definita durante la progettazione fisica della base di dati.

## Campionamento e filtraggio (data subsetting)

Come già accennato a causa di problemi legati alle risorse hardware, non è stato possibile utilizzare l'intera raccolta dati, pertanto si è scelto di operare un campionamento dei dati. Sono stati utilizzati degli script Python specifici che potessero elaborare i file CSV ottenuti precedentemente per estrarre dei sottoinsieme di dati consistenti e significativi.

Da un punto di vista prettamente tecnico il dataset iniziale è stato manipolato nel seguente modo: sono stati selezionati 10.000 utenti ed altrettante attività, mentre le restanti entità sono state estratte in maniera tale da garantire consistenza ed integrità dei dati.

Quando si parla di consistenza ed integrità si fa riferimento al fatto che gli script python "aggiuntivi" sono stati realizzati per prevenire eventuali errori di importazione come quelli che sarebbero potuti insorgere in caso di record orfani o vincoli di integrità referenziali violati.

## Importazione in Oracle

Una volta terminate le due fasi precedenti è stato eseguito il popolamento effettivo della base di dati sfruttando la procedura "Import Data" fornita dal software di gestione scelto.

# 5. ASPETTI FUNZIONALI DEL DB

## 5.1 Zona Query e Viste

```
-- QUERY E VISTE --
-- Individuare le attività con la valutazione media più alta in una città
SELECT Nome, Valutazione_Media
FROM ATTIVITA_COMMERCIALI
WHERE Città = 'Indianapolis'
AND Valutazione_Media =
(
    SELECT MAX(Valutazione_Media)
    FROM ATTIVITA_COMMERCIALI
    WHERE Città = 'Indianapolis'
);

-- Calcolare il numero medio di recensioni per categoria
CREATE MATERIALIZED VIEW Numero_Recensioni_Per_Categoria AS
SELECT AP.Categoria, COUNT(R.ID_Recensione) AS Num_Recensioni
FROM RECENSIONI R
JOIN APPARTENENZE_CATEGORIE AP ON R.Attività = AP.Attività
GROUP BY AP.Categoria;

-- Query per ottenere la media totale
SELECT Categoria, AVG(Num_Recensioni) AS Numero_Recensioni_Medio_Per_Categoria
FROM Numero_Recensioni_Per_Categoria
GROUP BY Categoria;
```

```

-- Determinare gli utenti più attivi
-- Utenti con il massimo numero di recensioni:
SELECT ID_Utente, Nome, Num_Recensioni
FROM UTENTI
ORDER BY Num_Recensioni DESC
FETCH FIRST 10 ROWS ONLY;
-- Utenti per numero totale di complimenti:
SELECT ID_Utente, Nome, (Num_Complimenti_Cool + Num_Complimenti_Funny + Num_Complimenti_Useful) AS Totale_Complimenti
FROM UTENTI
ORDER BY Totale_Complimenti DESC
FETCH FIRST 10 ROWS ONLY;

-- Analizzare la distribuzione temporale dei check-in
SELECT
    TO_CHAR(Data, 'Month', 'NLS_DATE_LANGUAGE = ITALIAN') AS Mese,
    COUNT(*) AS Numero_di_Check_In
FROM CHECK_IN
GROUP BY
    TO_CHAR(Data, 'Month', 'NLS_DATE_LANGUAGE = ITALIAN'),
    TO_CHAR(Data, 'MM')
ORDER BY
    TO_CHAR(Data, 'MM') DESC;

```

```

-- Individuare le categorie di attività con maggior densità di fotografie
CREATE OR REPLACE VIEW FOTO_PER_ATTIVITA_CATEGORIA AS
SELECT AC.Categoria, AC.Attivita, COUNT(F.ID_Fotografia) AS Num_Fotografie
FROM APPARTENENZE_CATEGORIE AC
LEFT JOIN FOTOGRAFIE F ON F.Attivita = AC.Attivita
GROUP BY AC.Categoria, AC.Attivita;

SELECT Categoria, AVG(Num_Fotografie) AS Densita_Foto
FROM FOTO_PER_ATTIVITA_CATEGORIA
GROUP BY Categoria
ORDER BY Densita_Foto ASC;

```

## 5.2 Zona Trigger

Per soddisfare i requisiti inerenti all'integrità dei dati e all'aggiornamento automatico delle metriche (come richiesto dalla traccia), sono stati implementati dei Trigger PL/SQL che vadano a rendere automatiche tutte quelle operazioni di aggiornamento a seguito di variazioni avvenute nel DB.

## Trigger per la Standardizzazione dei dati:

Abbiamo creato dei Trigger del tipo BEFORE INSERT per le tabelle ATTIVITA'\_COMMERCIALI , UTENTI , RECENSIONI E SUGGERIMENTI in maniera tale che il sistema vada in automatico a controllare la correttezza dei dati che si stanno aggiungendo gestendo i valori di default , così facendo impediamo la presenza di valori NULL laddove non ci dovrebbe essere

```
-- COSTRUZIONE DEI TRIGGER --
-- Standardizzazione dei dati
-- Trigger per la tabella ATTIVITA
CREATE TRIGGER STANDARD_ATTIVITA
BEFORE INSERT ON ATTIVITA_COMMERCIALI
FOR EACH ROW
BEGIN
    IF :NEW.Num_Recensioni IS NULL THEN :NEW.Num_Recensioni := 0; END IF;
    IF :NEW.Valutazione_Media IS NULL THEN :NEW.Valutazione_Media := 0; END IF;
END;

-- Trigger per la tabella UTENTI
CREATE TRIGGER STANDARD_UTENTI
BEFORE INSERT ON UTENTI
FOR EACH ROW
BEGIN
    IF :NEW.Data_Iscrizione IS NULL THEN :NEW.Data_Iscrizione := SYSDATE; END IF;
    IF :NEW.Num_Recensioni IS NULL THEN :NEW.Num_Recensioni := 0; END IF;
    IF :NEW.Num_Complimenti_Cool IS NULL THEN :NEW.Num_Complimenti_Cool := 0; END IF;
    IF :NEW.Num_Complimenti_Funny IS NULL THEN :NEW.Num_Complimenti_Funny := 0; END IF;
    IF :NEW.Num_Complimenti_Useful IS NULL THEN :NEW.Num_Complimenti_Useful := 0; END IF;
    IF :NEW.Valutazione_Media_Recensioni IS NULL THEN :NEW.Valutazione_Media_Recensioni := 0; END IF;
END;

-- Trigger per la tabella RECENSIONI
CREATE TRIGGER STANDARD_RECENSIONI
BEFORE INSERT ON RECENSIONI
FOR EACH ROW
BEGIN
    IF :NEW.Data IS NULL THEN :NEW.Data := SYSDATE; END IF;
    IF :NEW.Num_Valutazioni_Divertenti IS NULL THEN :NEW.Num_Valutazioni_Divertenti := 0; END IF;
    IF :NEW.Num_Valutazioni_Interessanti IS NULL THEN :NEW.Num_Valutazioni_Interessanti := 0; END IF;
    IF :NEW.Num_Valutazioni_Utili IS NULL THEN :NEW.Num_Valutazioni_Utili := 0; END IF;
END;

-- Trigger per la tabella SUGGERIMENTI
CREATE TRIGGER STANDARD_SUGGERIMENTI
BEFORE INSERT ON SUGGERIMENTI
FOR EACH ROW
BEGIN
    IF :NEW.Num_Complimenti IS NULL THEN :NEW.Num_Complimenti := 0; END IF;
END;
```

## Trigger per il ricalcolo automatico dei valori statistici:

I seguenti Trigger agiscono negli scenari di INSERT , UPDATE o DELETE sulla tabella RECENSIONI per assicurarsi di ricalcolare i valori delle medie espressi nelle relazioni stesse:

- **Nuova Recensione:** Aggiorna la media pesata includendo il nuovo voto e incrementa il contatore.
- **Modifica Voto:** Sottrae il peso del vecchio voto dalla media attuale e aggiunge quello nuovo, senza alterare il contatore.
- **Cancellazione:** Rimuove il peso del voto eliminato e decrementa il contatore.

```
-- Aggiornamento automatico della valutazione media di un'attività al momento dell'inserimento di una nuova recensione
CREATE OR REPLACE TRIGGER AGGIORNAMENTO_AUTOMATICO_ATTIVITA
AFTER INSERT OR UPDATE OR DELETE ON RECENSIONI
FOR EACH ROW
DECLARE
    -- Variabili di appoggio per recuperare lo stato corrente dal database
    Valutazione_Media_Attuale ATTIVITA_COMMERCIALI.Valutazione_Media%TYPE;
    Numero_Recensioni_Attuale ATTIVITA_COMMERCIALI.Num_Recensioni%TYPE;
BEGIN
    -- CASO 1: INSERIMENTO DI UNA NUOVA RECENSIONE
    IF INSERTING THEN
        UPDATE ATTIVITA_COMMERCIALI
        SET Valutazione_Media = ((Valutazione_Media * Num_Recensioni) + :NEW.Valutazione) / (Num_Recensioni + 1),
            Num_Recensioni = Num_Recensioni + 1
        WHERE ID_Attivita = :NEW.Attivita;

    -- CASO 2: MODIFICA DI UNA RECENSIONE ESISTENTE
    ELSIF UPDATING THEN
        -- Leggiamo i valori attuali
        SELECT Valutazione_Media, Num_Recensioni
        INTO Valutazione_Media_Attuale, Numero_Recensioni_Attuale
        FROM ATTIVITA_COMMERCIALI
        WHERE ID_Attivita = :NEW.Attivita
        FOR UPDATE; -- Per la gestione della concorrenza delle transazioni
        -- se ci sono 0 recensioni, non facciamo nulla per evitare divisioni per zero
        IF Numero_Recensioni_Attuale > 0 THEN
            UPDATE ATTIVITA_COMMERCIALI
            SET Valutazione_Media = ((Valutazione_Media_Attuale * Numero_Recensioni_Attuale) - :OLD.Valutazione + :NEW.Valutazione) / Numero_Recensioni_Attuale
            WHERE ID_Attivita = :NEW.Attivita;
        END IF;

    -- CASO 3: CANCELLAZIONE DI UNA RECENSIONE ESISTENTE
    ELSIF DELETING THEN
        -- Recupero stato attuale
        SELECT Valutazione_Media_Recensioni, Num_Recensioni
        INTO Valutazione_Media_Recensioni_Attuale, Numero_Recensioni_Utente_Attuale
        FROM UTENTI
        WHERE ID_Utente = :OLD.Utente
        FOR UPDATE;

        IF Numero_Recensioni_Utente_Attuale <= 1 THEN
            -- Reset totale se era l'ultima recensione
            UPDATE UTENTI
            SET Valutazione_Media_Recensioni = 0,
                Num_Recensioni = 0
            WHERE ID_Utente = :OLD.Utente;
        ELSE
            UPDATE UTENTI
            SET Valutazione_Media_Recensioni = ((Valutazione_Media_Recensioni_Attuale * Numero_Recensioni_Utente_Attuale) - :OLD.Valutazione) / (Numero_Recensioni_Utente_Attuale - 1),
                Num_Recensioni = Numero_Recensioni_Utente_Attuale - 1
            WHERE ID_Utente = :OLD.Utente;
        END IF;

    END IF;
END;
```

```

-- Aggiornamento automatico della Valutazione_Media_Recensioni e Num_Recensioni in UTENTI

CREATE OR REPLACE TRIGGER AGGIORNAMENTO_AUTOMATICO_UTENTE
AFTER INSERT OR UPDATE OR DELETE ON RECENSIONI
FOR EACH ROW
DECLARE
    -- Variabili per leggere lo stato attuale dal database
    Valutazione_Media_Recensioni_Attuale UTENTI.Valutazione_Media_Recensioni%TYPE;
    Numero_Recensioni_Utente_Attuale UTENTI.Num_Recensioni%TYPE;
BEGIN
    -- CASO 1: INSERIMENTO DI UNA NUOVA RECENSIONE
    IF INSERTING THEN
        UPDATE UTENTI
        SET Valutazione_Media_Recensioni = ((Valutazione_Media_Recensioni * Num_Recensioni) + :NEW.Valutazione) / (Num_Recensioni + 1),
            Num_Recensioni = Num_Recensioni + 1
        WHERE ID_Utente = :NEW.Utente;

    -- CASO 2: MODIFICA DI UNA RECENSIONE
    ELSIF UPDATING THEN
        -- Leggiamo i valori attuali
        SELECT Valutazione_Media_Recensioni, Num_Recensioni
        INTO Valutazione_Media_Recensioni_Attuale, Numero_Recensioni_Utente_Attuale
        FROM UTENTI
        WHERE ID_Utente = :NEW.Utente
        FOR UPDATE; -- Per la gestione della concorrenza delle transazioni
        -- se l'utente non aveva precedentemente scritto alcuna recensione , il DB non fa niente per evitare errori(n. recensioni = 0)
        IF Numero_Recensioni_Utente_Attuale > 0 THEN
            UPDATE UTENTI
            SET Valutazione_Media_Recensioni = ((Valutazione_Media_Recensioni_Attuale * Numero_Recensioni_Utente_Attuale) - :OLD.Valutazione + :NEW.Valutazione) / Numero_Recensioni_Utente_Attuale
            WHERE ID_Utente = :NEW.Utente;
        END IF;

    -- CASO 3: CANCELLAZIONE DI UNA RECENSIONE ESISTENTE
    ELSIF DELETING THEN
        -- Recupero stato attuale
        SELECT Valutazione_Media_Recensioni, Num_Recensioni
        INTO Valutazione_Media_Recensioni_Attuale, Numero_Recensioni_Utente_Attuale
        FROM UTENTI
        WHERE ID_Utente = :OLD.Utente
        FOR UPDATE;

        IF Numero_Recensioni_Utente_Attuale <= 1 THEN
            -- Reset totale se era l'ultima recensione
            UPDATE UTENTI
            SET Valutazione_Media_Recensioni = 0,
                Num_Recensioni = 0
            WHERE ID_Utente = :OLD.Utente;
        ELSE
            UPDATE UTENTI
            SET Valutazione_Media_Recensioni = ((Valutazione_Media_Recensioni_Attuale * Numero_Recensioni_Utente_Attuale) - :OLD.Valutazione) / (Numero_Recensioni_Utente_Attuale - 1),
                Num_Recensioni = Numero_Recensioni_Utente_Attuale - 1
            WHERE ID_Utente = :OLD.Utente;
        END IF;

    END IF;

END;

```

## Trigger per la verifica dei check-in:

Il seguente trigger agisce a seguito di aggiunte sulla tabella CHECK-IN assicurandosi che gli utenti non possano aggiungere check-in la cui data superi quella odierna(SYSDATE) per evitare una falla logica del DB.

```
-- Un check-in non può avere una data maggiore di SYSDATE
CREATE OR REPLACE TRIGGER VALIDAZIONE_DATA_CHECK_IN
BEFORE INSERT OR UPDATE ON CHECK_IN
FOR EACH ROW
BEGIN
    -- Controllo: Se la data inserita è maggiore di adesso
    IF :NEW.Dats > SYSDATE THEN
        RAISE_APPLICATION_ERROR(-20001, 'Errore: Non è possibile inserire check-in con data futura.');
    END IF;
END;
```

## 5.3 Zona Stored Procedure

*Controllo sulle modifiche dei contenuti da parte degli utenti:*

Abbiamo creato 3 procedure che andassero (per i tre contenuti pubblicabili da un utente) a verificare la legittimità dell'operazione , ovvero che l'utente che stia provando ad effettuare la modifica sia l'effettivo autore del contenuto e che dunque abbia i giusti permessi per eseguire la modifica stessa.

```
-- DEFINIZIONE STORED PROCEDURE --
-- L'utente non può modificare le recensioni altrui
CREATE OR REPLACE PROCEDURE ABILITA_MODIFICA_RECENSIONE (
    -- Variabili della procedura di input(no modifica)
    ID_Recensione_Procedura IN VARCHAR2,
    Valutazione_Procedura IN INTEGER,
    Testo_Procedura IN CLOB,
    Utente_Procedura IN VARCHAR2
) AS
    Proprietario_Recensione VARCHAR2(30);
BEGIN
    -- Trova chi è il proprietario della recensione
    SELECT Utente INTO Proprietario_Recensione
    FROM RECENSIONI
    WHERE ID_Recensione = ID_Recensione_Procedura;

    -- Se l'utente corrisponde, esegui l'update
    IF Proprietario_Recensione = Utente_Procedura THEN
        UPDATE RECENSIONI
        SET Valutazione = Valutazione_Procedura,
            Testo = Testo_Procedura,
            Data = SYSDATE
        WHERE ID_Recensione = ID_Recensione_Procedura;
        COMMIT;
    ELSE
        RAISE_APPLICATION_ERROR(-20001, 'Errore: Non puoi modificare recensioni altrui.');
    END IF;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20002, 'Recensione non trovata.');
END;
```

```

-- L'utente non può modificare i suggerimenti altrui
CREATE OR REPLACE PROCEDURE ABILITA_MODIFICA_SUGGERIMENTO (
    Data_Procedura IN DATE,
    Attivita_Procedura IN VARCHAR2,
    Testo_Procedura IN CLOB,
    Utente_Procedura IN VARCHAR2
) AS
    Contatore INTEGER;
BEGIN
    -- Controllo se esiste il record per questo utente
    SELECT COUNT(*)
    INTO Contatore
    FROM SUGGERIMENTI
    WHERE Data = Data_Procedura
        AND Attivita = Attivita_Procedura
        AND Utente = Utente_Procedura;

    IF Contatore > 0 THEN
        -- Il record esiste ed è dell'utente: Esegue l'Update
        UPDATE SUGGERIMENTI
        SET Testo = Testo_Procedura
        WHERE Data = Data_Procedura
            AND Attivita = Attivita_Procedura
            AND Utente = Utente_Procedura;

        COMMIT;
    ELSE
        RAISE_APPLICATION_ERROR(-20002, 'Errore: Non puoi modificare i suggerimenti altrui.');
    END IF;
END;
/

-- Abilita la modifica della fotografia solo al proprietario
CREATE OR REPLACE PROCEDURE ABILITA_MODIFICA_FOTOGRAFIA(
    ID_Fotografia_Procedura IN VARCHAR2,
    Didascalia_Procedura IN CLOB,
    Etichetta_Procedura IN VARCHAR2,
    Utente_Procedura IN VARCHAR2
) AS
    Proprietario_Fotografia VARCHAR2(30);
BEGIN
    -- Trova chi è il proprietario della fotografia
    SELECT Utente INTO Proprietario_Fotografia
    FROM FOTOGRAFIE
    WHERE ID_Fotografia = ID_Fotografia_Procedura;

    -- Se l'utente corrisponde, esegui l'update
    IF Proprietario_Fotografia= Utente_Procedura THEN
        UPDATE FOTOGRAFIE
        SET Testo = Didascalia_Procedura,
            Etichetta = Etichetta_Procedura
        WHERE ID_Fotografia = ID_Fotografia_Procedura;

        COMMIT;
    ELSE
        RAISE_APPLICATION_ERROR(-20001, 'Errore: Non puoi modificare fotografie altrui.');
    END IF;

    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            RAISE_APPLICATION_ERROR(-20002, 'Fotografia non trovata.');
END;
/

```

```

-- L'amico deve essere sempre diverso dall'utente e se già è stato aggiunto non può essere nuovamente inserito
CREATE OR REPLACE PROCEDURE AGGIUNGI_AMICO(
    Utente_Procedura_User IN VARCHAR2,
    Utente_Procedura_Amico IN VARCHAR2
) AS
    Conteggio INTEGER;
BEGIN
    -- Controllo se sono lo stesso utente
    IF Utente_Procedura_Amico = Utente_Procedura_User THEN
        RAISE_APPLICATION_ERROR(-20001, 'Errore: Non puoi aggiungere te stesso come amico.');
    END IF;

    -- Controllo se l'amicizia esiste già
    SELECT COUNT(*)
    INTO Conteggio
    FROM AMICIZIE
    WHERE Utente = Utente_Procedura_User
    AND Amico = Utente_Procedura_Amico;

    IF Conteggio > 0 THEN
        RAISE_APPLICATION_ERROR(-20001, 'Errore: Non puoi inserire nuovamente questo amico.');
    ELSE

        INSERT INTO AMICIZIE (Utente, Amico)
        VALUES (Utente_Procedura_User, Utente_Procedura_Amico);

        COMMIT;
    END IF;
END;
/

```

## 6. OTTIMIZZAZIONE DEL DB

### 6.1 Indici

Per ottimizzare le varie query inserite nel nostro progetto, abbiamo deciso di implementare l'utilizzo di vari indici che riportiamo qui di sotto.

**NOTA:** In ogni foto è riportato per ogni indice il suo utilizzo pratico e per quale query ottimizza i tempi

```

-- INDICI PER OTTIMIZZARE LA RICERCA IN QUERY E VISTE--
CREATE INDEX RECENSIONI_ATTIVITA_INDICE ON RECENSIONI(Attivita);
CREATE INDEX RECENSIONI_UTENTE_INDICE ON RECENSIONI(Utente);
CREATE INDEX FOTOGRAFIE_ATTIVITA_INDICE ON FOTOGRAFIE(Attivita);

-- Ottimizzazione pensata per la query che richiede il calcolo del num. medio di recensioni per categoria
CREATE INDEX APPARTENZE_CATEGORIE_ATTIVITA_INDICE ON APPARTENENZE_CATEGORIE(Attivita);

-- Ottimizzazione pensata per la query che necessita l'esecuzione di un ricerca per città
CREATE INDEX CITTA_INDICE ON ATTIVITA_COMMERCIALI(Città);

-- Ottimizzazione pensata per migliorare l'analisi della distribuzione temporale dei check-in
CREATE INDEX CHECKIN_DATA_INDICE ON CHECK_IN(Data);

-- Ottimizzazione pensata per migliorare le query che restituiscono una classifica
CREATE INDEX VALUTAZIONE_INDICE ON ATTIVITA_COMMERCIALI(Valutazione_Media DESC);
CREATE INDEX NUM_RECENSIONI_INDICE ON UTENTI(Num_Recensioni DESC);

```

## 6.2 Gestione della concorrenza

In una piattaforma social come quella progettata, il rischio di incorrere in delle transazioni concorrenti che modifichino la base di dati causando gravi problemi di integrità e consistenza risulta essere molto elevato.

Per questo motivo si è deciso di adottare un controllo della concorrenza pessimistico come il Two-Phase Locking Strict (2PL Stretto), grazie al quale la gestione dei lock sugli oggetti della base di dati è tale da evitare anche anomalie di dirty read (lettura sporca).

*La versione standard del protocollo 2PL fa sì che ogni transazione sia coinvolta in due fasi:*

- 1) **La prima fase**, detta fase crescente, prevede che la transazione acquisisca in massa tutti permessi necessari a sbloccare i lock che “proteggono” gli oggetti con cui dovrà lavorare
- 2) **La seconda fase**, detta fase decrescente, prevede che successivamente allo svolgimento dell’intera sequenza di operazioni, la transazione liberi ogni oggetto mediante l’opportuna primitiva

Invece la versione adottata, 2PL Strict, per lo sviluppo della piattaforma social prevede che i lock vengano mantenuti fino al commit o abort della transazione.

*Volendo motivare e contestualizzare è possibile affermare che:*

- Gli utenti della piattaforma potranno inserire recensioni, suggerimenti, fotografie ed eventuali check-in in maniera concorrente
- I dettagli di tipo “statistico” saranno sempre aggiornati in maniera coerente rispetto alle informazioni che sono presenti e/o che verranno inseriti nella base di dati

## 6.3 Gestione dell'affidabilità

A valle di malfunzionamenti di tipo software o hardware ogni DBMS deve essere ben equipaggiato con un Gestore dell'affidabilità che permetta di ripristinare, qualora fosse necessario, il corretto stato dell'intera base di dati.

Il problema principale sussiste con le transazioni che prevedono operazioni di scrittura, dato che esse non sono a tutti gli effetti delle transazioni atomiche. Pertanto gli effetti di una transazione di questo tipo, con il commit in memoria primaria potrebbero essere non riportati subito in memoria secondaria e dunque non essere persistenti.

Affinchè il Gestore dell'affidabilità possa maneggiare l'esecuzione di tutte le transazioni tenendo conto di commit o rollback in maniera persistente, esso necessita di uno strumento chiamato file di log, ovvero un file presente su memoria stabile che registra tutte le operazioni svolte dalle transazioni nel loro ordine di esecuzione.

Il file di log non è altro che un diario di bordo che ci permette di ricostruire in maniera veloce il corretto contenuto della base di dati dopo ciascuna transazione effettuata.

### Strategia di backup:

Il gestore dell'affidabilità utilizza dei precisi strumenti per attuare eventuali azioni di ripristino:

- **Checkpoint:** Consiste nella registrazione nel file di log dello stato delle transazioni attive in un determinato istante. Questo lavoro indotto dal DBMS riduce il tempo necessario per il ripristino del sistema, poiché parte delle modifiche apportate alla base di dati già sono state consolidate sul disco e non semplicemente salvate in memoria
- **Dump:** Consiste in una fotografia dello stato del database e pertanto funge da eventuale punto di partenza per un qualsiasi ripristino da effettuare in seguito a gravi problemi come il danneggiamento della memoria secondaria

## **Strategia di recovery:**

Le azioni di recovery mirano a ricostruire la base di dati sull'analisi dei file di log a partire dall'ultimo record dump disponibile, usufruendo delle azioni di undo e redo.

Per una piattaforma social con un'infrastruttura articolata come quella progettata, prima di agire bisogna valutare la gravità del guasto dimodochè il sistema possa ritornare subito operativo garantendo una perdita di dati minima:

- Se il guasto è di tipo “**soft**”, quindi relativo ad errori di programma o crash di sistema, allora si perde il contenuto della sola memoria centrale mentre viene conservato il contenuto della memoria stabile. In condizioni del genere è necessaria una ripresa a caldo (warm restart).
- In caso di guasti “**hard**”, pertanto malfunzionamento relativi alle memoria di massa, si perdono anche i dati memorizzati sullo storage secondario. Tale perdita implica l'utilizzo di una ripresa a freddo (cold restart).

# 7. INTERFACCIA APEX

## 7.1 Integrazione con il Database

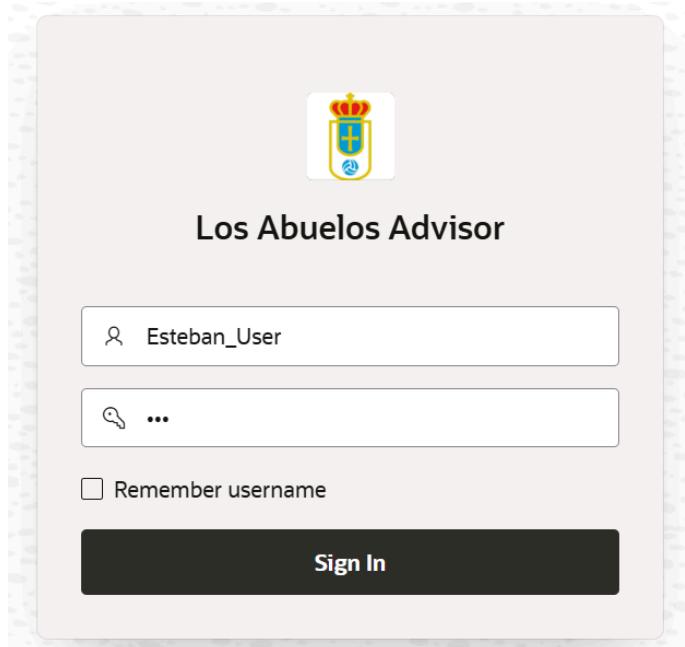
Per la realizzazione dell'interfaccia utente abbiamo utilizzato **Oracle APEX**. Questa piattaforma ci ha permesso di creare un'applicazione web collegata direttamente al database Oracle sviluppato nei capitoli precedenti.

Il vantaggio principale di questa scelta è stato poter richiamare direttamente le nostre tabelle, le Viste e il codice PL/SQL (Procedure e Trigger) senza dover creare complessi strati intermedi di connessione.

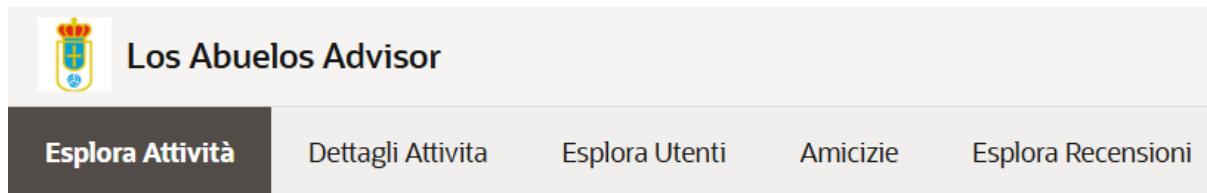
Inoltre, abbiamo sfruttato gli strumenti di sicurezza nativi di APEX per gestire i diversi permessi tra utenti standard, moderatori e amministratori, rispecchiando i ruoli definiti nella progettazione fisica.

## 7.2 Accesso e Struttura di Navigazione

L'applicazione inizia con la pagina di **Login**, un passaggio fondamentale per identificare l'utente e assegnare i privilegi corretti (Admin, Moderator o User). Il sistema verifica le credenziali inserite confrontandole con quelle memorizzate nel database e avvia la sessione appropriata.



Una volta effettuato l'accesso, l'elemento centrale che guida l'esperienza utente è la **Barra di Navigazione**, visibile in ogni pagina della piattaforma per garantire un orientamento costante.



Come mostrato in figura, il menu è stato organizzato per rendere immediate le operazioni principali.

Da qui è possibile accedere rapidamente a tutte le sezioni che analizzeremo nel dettaglio nei paragrafi successivi: dall'esplorazione delle **Attività Commerciali** alla gestione della rete sociale in **Esplora Utenti**, con la possibilità di aggiungere un amico e visualizzarlo nella sezione **Amicizie**, fino alla consultazione globale delle **Recensioni**.

Questa struttura modulare permette all'utente di muoversi agilmente tra le diverse funzionalità offerte dal database senza mai perdere il contesto di navigazione.

### 7.3 Home Page: Esplora Attività

La pagina principale dell'applicazione è denominata "**Esplora Attività**".

Come suggerisce il nome, questa schermata permette all'utente di visualizzare l'elenco completo di tutte le attività commerciali registrate nel nostro database.

A screenshot of the "Esplora Attività" search results page. At the top left is a search bar with the placeholder "Search...". To its right are filters for "Valutazione Media" (Rating) and "Stato" (Status). The main content area shows a list of businesses with their names, addresses, and average ratings. The first result is "&amp;pizza - Willow Grove" located at 2500 W Moreland Rd, Ste 3041, with a rating of 4.5. The second result is "026 Bar and Grill" located at 515 Gravois Rd, with a rating of 3.5. A "Total Row Count 10K" message is displayed above the results. A "Reset" button is located in the top right corner of the search area.

*L'interfaccia è stata progettata per essere intuitiva e funzionale:*

sulla sinistra è presente un pannello per filtrare i risultati in base alle preferenze, mentre la vista centrale recupera i dati in tempo reale dalla tabella ATTIVITA\_COMMERCIALI.

Ogni scheda attività riporta immediatamente informazioni essenziali come l'indirizzo e la votazione media, garantendo che qualsiasi nuova attività o modifica nel backend sia subito visibile agli utenti senza ritardi.

## 7.4 Scheda Dettaglio e Inserimento Recensioni

Selezionando una specifica attività dalla lista, si accede alla pagina di **Dettaglio**.

*Questa schermata è il cuore dell'interazione utente:*

qui vengono mostrate tutte le informazioni approfondite relative al locale selezionato (Indirizzo, valutazione, descrizione, ecc...), recuperate tramite query puntuali sul database.

The screenshot shows the 'Dettagli Attività' (Activity Details) page for the establishment 'Butter Burger'. The top navigation bar includes links for 'Espora Attività', 'Dettagli Attività' (which is active and highlighted in dark grey), 'Espora Utenti', 'Amicizie', and 'Espora Recensioni'. A user profile icon for 'esteban\_user' is also present. The main content area displays the following details:

- Nome:** Butter Burger
- Valutazione Media:** ★★★★☆ (4.5 stars)
- Indirizzo:** Via Roma 10
- Città:** Pomigliano
- Cap:** 20121
- Stato:** IT

Below these fields, there are two sections for reviews:

- STAMPS\_MODERATOR** (1/26/2026):  
Locale pulito e burger saporito. Consigliato per i controlli di qualità!
- STAMPS\_MODERATOR** (1/26/2026):  
Questo panino è la mia vita, non esiste nulla al mondo di più buono

A large text input field at the bottom contains the placeholder text "Outside" and "Che locale eccezionale !!". A black button labeled "Scrivi Recensione" (Write Review) is located at the bottom right of the review section.

Oltre alla consultazione, questa pagina permette all'utente di interagire attivamente. Cliccando sul pulsante "Scrivi Recensione", si apre la maschera di inserimento dedicata:

The screenshot shows a modal window titled "Scrivi una recensione". Inside, there's a "Data" field containing "1/20/2026" with a calendar icon. A "Testo" section contains the text: "Quella pizza con l'ananas sopra era un insulto a tutti i napoletani. Vergognatevi !!!". Below it is a "Valutazione" section with a 1-star rating and a "Required" label. An "Utente" field contains "Scott\_8\_McFratm" with a "Required" label. At the bottom are "Cancel", "Delete", and "Apply Changes" buttons.

Come si vede in figura, il form è essenziale e richiede all'utente di assegnare un voto (da 1 a 5) e un commento testuale.

*Il salvataggio di questi dati non è una semplice operazione di scrittura: il pulsante di conferma attiva infatti i Trigger definiti nel Capitolo 5 (paragrafo 5.2). In questo modo, appena un utente pubblica il voto, il sistema ricalcola automaticamente la media complessiva dell'attività e aggiorna le statistiche del profilo utente, garantendo la consistenza dei dati in tempo reale.*

## 7.5 Gestione Social: Esplora Utenti

Essendo il progetto una piattaforma social, abbiamo dedicato una sezione specifica alla gestione delle relazioni tra gli iscritti, denominata "**Esplora Utenti**".

Questa pagina offre una panoramica di tutti gli utenti registrati nel database, presentandoli tramite delle card riassuntive.

The screenshot shows a web application interface for 'Los Abuelos Advisor'. At the top, there's a navigation bar with links: 'Esplora Attività', 'Dettagli Attività', 'Esplora Utenti' (which is highlighted in black), 'Amicizie', and 'Esplora Recensioni'. A search bar with placeholder text 'Ricerca per nome:' and a 'Go' button are also at the top. Below the navigation, there's a section titled 'Order By' with a dropdown menu set to 'Nome'. The main content area displays a grid of user cards. Each card contains the user's initials in a colored box, their name, the date of registration, and the number of reviews they have written. There are also buttons labeled 'Aggiungi Amico' for each user. The users listed are: Jennifer (JE, 11/9/2009, 6679 reviews), Chris (CH, 3/8/2010, 2531 reviews), Mary Kate (MK, 6/4/2010, 2360 reviews), Matt (MA, 10/24/2006, 2227 reviews), Doreen (DO, 12/4/2009, 1758 reviews), Mimi (MI, 3/30/2011, 1568 reviews), Blake (BL, 2/25/2008, 1302 reviews), Jason (JA, 7/31/2006, 1283 reviews), Debra (DE, 6/28/2016, 940 reviews), Sklar (SK, 8/21/2009, 907 reviews), Ocean (OC, 7/22/2007, 869 reviews), Chris (CH, 1/27/2010, 869 reviews), and Jada (JA, 7/30/2016, 857 reviews).

La funzionalità principale di questa schermata è la gestione delle amicizie. Ogni card utente presenta un pulsante dedicato per aggiungere la persona alla propria lista amici.

Anche in questo caso, l'azione sull'interfaccia richiama direttamente la logica del database: il click attiva la procedura di controllo descritta nel paragrafo 5.3 (Zona Stored Procedure), che verifica la validità della richiesta (ad esempio, impedendo di aggiungere se stessi o relazioni già esistenti) prima di inserire il record nella tabella AMICIZIE.

## 7.6 Il Mio Network: Pagina Amicizie

Una volta aggiunti degli utenti tramite la funzione "Esplora", è fondamentale poter gestire i propri contatti in modo ordinato.

Per questo abbiamo creato la pagina "**Amicizie**", pensata per mostrare all'utente loggato esclusivamente la lista delle persone con cui ha già stretto amicizia.

The screenshot shows a user interface for managing social connections. At the top, there's a navigation bar with links: 'Los Abuelos Advisor', 'esteban\_user', 'Escopra Attività', 'Dettagli Attività', 'Escopra Utenti', 'Amicizie' (which is highlighted in dark grey), and 'Escopra Recensioni'. Below the navigation is a search bar labeled 'Order By' with the dropdown menu open, showing 'Nome'. The main content area displays a single friend's profile card. The card features a small blue icon with 'AC', the name 'Antonio Conte', the date '1/26/2026', and a small number '0' indicating no reviews. Below the card, it says '0 Recensioni'.

A differenza della ricerca globale, questa vista filtra automaticamente i risultati basandosi sull'ID dell'utente connesso, recuperando dalla tabella AMICIZIE solo le relazioni confermate.

Questa suddivisione permette di monitorare facilmente il proprio grafo sociale all'interno della piattaforma, tenendo separata la fase di ricerca (Esplora Utenti) da quella di consultazione dei propri legami.

## 7.7 Consultazione Globale: Esplora Recensioni

Per offrire una visione d'insieme sui contenuti della piattaforma, abbiamo creato la pagina "**Esplora Recensioni**".

A differenza della scheda dettaglio (che filtra per singola attività), questa schermata interroga l'intera tabella RECENSIONI, permettendo all'utente di sfogliare tutti i feedback presenti nel database.

The screenshot shows a search interface for reviews. At the top, there's a search bar with a magnifying glass icon, a 'Go' button, and an 'Actions' dropdown menu. Below the search bar is a table header with columns: 'Id Recensione', 'Data', 'Testo', 'Valutazione', 'Num Valutazioni Divertenti', 'Num Valutazioni Interessanti', 'Num Valutazioni Utili', 'Attività', and 'Utente'. The table body contains three rows of review data. The first row has an ID of '7YYvUwScjZ70kJVM2-kbJw', a date of '4/15/2013', a text entry 'Decent pizza. A little pricey. its in an airport so what do you expect?', a value of '2', zero divertenti, zero interessanti, zero utili, an activity ID 'XXZqNGRUyGVlx8oNm6BtBA', and a user ID 'pDlf2NtkXaBzY-eEMVo35g'. The second row has an ID of 'AbxOlf-6Bhc66iTsuKZu2A', a date of '9/29/2011', a text entry 'Nice cozy place with a decent beet and wine selection. Definately interested in coming back to the place, and as a completely strange compliment the counter tops rock.', a value of '4', zero divertenti, zero interessanti, zero utili, an activity ID 'JOvNunKJlWhwJRwlMmUR9ZA', and a user ID 'knZRTqlgxXmJyvQBa9f-A'. The third row has an ID of 'N9GPG2avGMSAKxpOmWF5w', a date of '12/23/2015', a text entry 'Very rude 'managers' who threatened to sue me when I said I'd leave them a bad review... Can you believe that? Do NOT stay here!', a value of '1', zero divertenti, zero interessanti, zero utili, an activity ID 'n5P6-HWN3kFSYBxFfDxEg', and a user ID 'Qq6n\_6EPO\_oUNn7ylhnnww'.

L'interfaccia permette di esplorare i commenti lasciati dalla community e, selezionando una voce specifica, di visualizzarne tutti i relativi dettagli (come la data, l'autore e la valutazione). Questa funzionalità è utile per dare risalto alle opinioni degli utenti indipendentemente dal locale visitato, facilitando la scoperta di nuove esperienze.

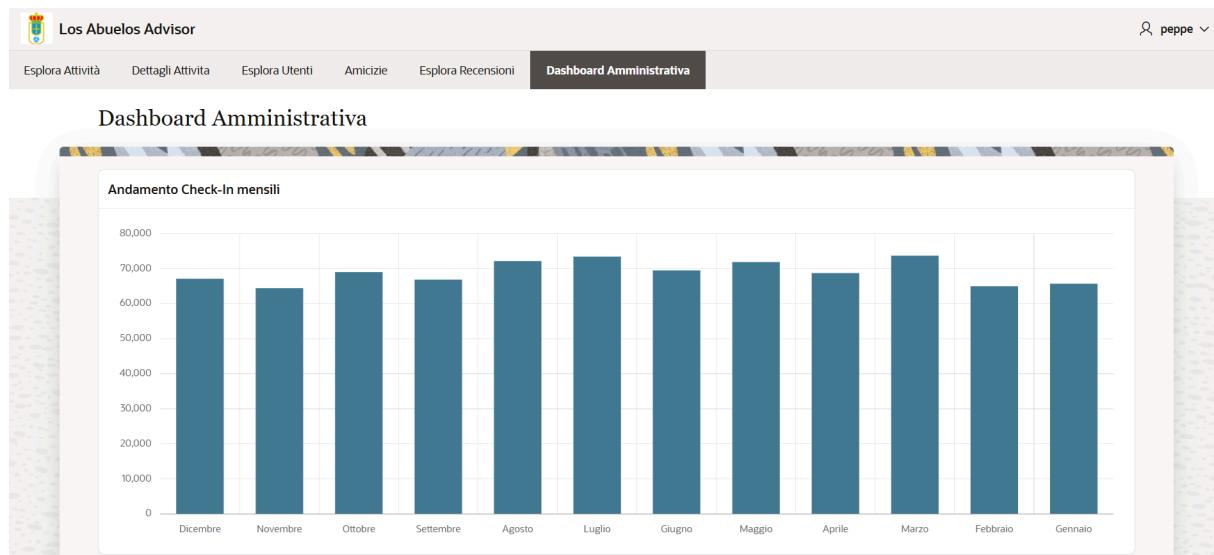
## 7.8 Dashboard Amministrativa e Statistiche

L'ultima parte dell'interfaccia è riservata esclusivamente agli utenti con ruolo **Admin**.

Grazie agli Authorization Schemes configurati in APEX, questa sezione di monitoraggio rimane nascosta agli utenti standard e ai moderatori.

La Dashboard è stata progettata per offrire una sintesi visiva dell'andamento della piattaforma attraverso grafici interattivi. In particolare, abbiamo implementato:

- Un grafico temporale che mostra l'andamento dei **check-in mensili**, utile per identificare i periodi di maggiore affluenza.



- Due classifiche ("Top 10") degli **utenti più attivi**, basata sul numero di contributi, interazioni e recensioni.

The screenshot shows a dashboard titled "Los Abuelos Advisor". At the top, there are navigation links: "Esplora Attività", "Dettagli Attività", "Esplora Utenti", "Amicizie", "Esplora Recensioni", and "Dashboard Amministrativa". A search bar and a user profile icon are also at the top right. The main area contains two tables side-by-side:

Nome	Num Recensioni	Totale Complimenti
Jennifer	6679	47353
Chris	2531	1729
Mary Kate	2360	34189
Matt	2227	9453
Doreen	1758	9740
Mirri	1568	40356
Blake	1302	1235
Jason	1283	3352
Debra	940	3667
Sklar	907	5600

Below the first table: "1 - 10"

Nome	Totale Recensioni	Media Voti
Mary Kate	34	3.85
John	31	3.32
Will	25	3.44
Ryan	23	3.96
Abby	19	4.26
Renee	18	4.06
Courtney	15	4.13
Heather	15	4.27
Angela	14	4.64
Mitch	14	4.64

Below the second table: "1 - 10"

Questi strumenti analitici non memorizzano dati statici, ma vengono generati dinamicamente interrogando le **Viste (Views)** create nella sezione "Zona Query" del database (Capitolo 5).

Questo approccio garantisce che l'amministratore abbia sempre sotto controllo la situazione reale della community aggiornata all'ultimo secondo.

## **8. Esecuzione di query in linguaggio naturale**

Dati i recenti sviluppi nell'ambito dell'intelligenza artificiale è stato fatto un tentativo di implementazione di un cosiddetto modello di GenBI conosciuto con il nome di WrenAI.

Nel contesto della piattaforma realizzata, la possibilità di integrare un sistema di Text-to-SQL sarebbe ideale per supportare tutti quegli utenti “non tecnici” durante operazioni di ricerca riguardanti informazioni aggregate o statistiche.

L'implementazione di WrenAI riuscirebbe ad abbattere la barriera costituita dalla rigida sintassi del linguaggio SQL, rendendo l'esperienza generale della piattaforma molto più fluida ed “aperta” ad un qualsiasi tipo di utente, esperto o meno che esso sia.

Volendo contestualizzare maggiormente i vantaggi derivanti dall'utilizzo di WrenAI, basti pensare agli amministratori della piattaforma che in maniera flessibile e rapida potrebbero generare nuovi report che rispondano a domande del tipo:

- Qual è l'attività più popolare della piattaforma?
- Quali attività sono attualmente in voga?

## 8.1 Architettura di WrenAI

L'implementazione di WrenAI all'interno della piattaforma social progettata si basa su un'architettura stratificata ed organizzata per trasformare i dati grezzi provenienti dal database di turno in informazioni strutturate reperibili attraverso il linguaggio naturale

- DATA LAYER: Il fondamento dell'integrazione prevede la connessione diretta tra WrenAI e l'istanza del nostro Oracle Database. L'interazione avviene tramite l'interfaccia JDBC e tramite la concessione di un account con i privilegi di SELECT, CREATE VIEW e DROP VIEW
- SEMANTIC LAYER: Attraverso l'adozione di un Modeling Definition Language (MDL), lo schema ER viene tradotto in concetti di business comprensibili dall'intelligenza artificiale. Quindi durante questa fase l'AI deve essere istruita su concetti come quelli della "popolarità" o del "trendy".
- AGENTIC LAYER: L'agente AI elabora le richieste formulate dagli utenti in linguaggio naturale interrogando il Semantic Layer. Tramite la contestualizzazione precedentemente effettuata, la domanda posta viene tradotta in query SQL ottimizzate per il DBMS Oracle
- REPRESENTATION LAYER: Il risultato ricavato dalle query SQL generate viene reso fruibile per l'utente tramite la generazione automatica di tabelle e/o grafici che facilitino la lettura dell'informazione

## 8.2 Problemi riscontrati

Da un punto di vista tecnico, il deployment di WrenAI prevederebbe l'installazione tramite Docker e l'integrazione con un provider LLM.

Tuttavia, l'implementazione finale ha dovuto tenere conto di un fattore determinante:

- **Sostenibilità Economica:**  
L'attivazione delle API Key degli LLM necessari a rendere funzionante l'infrastruttura di WrenAI, avrebbe comportato costi di gestione non previsti.