

UNIVERSITÀ DEGLI STUDI DI PADOVA

CORSO DI LAUREA IN INFORMATICA

Base di Dati per un Sistema di gestione di Serie TV in Streaming

Gruppo: Ceron Tommaso, Parolin Dennis

Anno Accademico 2024/2025

Indice

1	Abstract	2
2	Analisi dei Requisiti	2
3	Progettazione concettuale	4
3.1	Lista entità	4
3.2	Lista relazioni	5
3.3	Lista generalizzazioni	7
3.4	Schema E-R	7
4	Progettazione Logica	7
4.1	Analisi delle ridondanze	8
4.1.1	Ridondanza "NumeroSerieTv"	8
4.1.2	Ridondanza "NumeroUtenti"	9
4.2	Eliminazione delle generalizzazioni	11
4.3	Schema relazionale	11
5	Implementazione in PostgreSQL e Definizione delle Query	13

1 Abstract

Questo progetto sviluppa una base di dati per gestire delle serie tv, composte da stagioni ed episodi, e dai loro attori e registi. Inoltre, questa base di dati permette di gestire degli utenti e gli episodi che guardano, ai quali possono dare un voto.

L'obiettivo principale è quello di gestire la struttura delle serie tv, gestire le relazioni di quest'ultime con il cast e il regista che vi partecipa e infine registrare ciò che un utente guarda.

La base di dati distingue le persone in utenti, registi e attori, ognuno con degli attributi e relazioni distinte.

Una serie si compone di almeno una stagione, che a sua volta è composta da almeno un episodio. Opzionalmente una Serie TV può anche contenere una Opening (ovvero una sigla iniziale).

Infine, il sistema tiene traccia delle piattaforme di streaming che mettono a disposizione una determinata serie tv, e degli utenti che sottoscrivono un abbonamento con uno o più di queste piattaforme.

2 Analisi dei Requisiti

In questa sezione riassumiamo i requisiti che caratterizzano la base di dati.

Serie TV. Una serie TV è identificata da un titolo e un anno e contiene questi attributi:

- Titolo
- Anno di Inizio
- Descrizione
- Genere

Una Serie TV è composta da una o più stagioni, e da zero o una Opening (sigla iniziale)

Stagioni. Una stagione è identificata dal suo numero e dalla Serie Tv a cui appartiene, e contiene questi attributi:

- Numero
- Anno

Una stagione è composta da uno o più episodi

Episodio. Un episodio è identificato dal suo numero e dalla stagione a cui appartiene, e contiene questi attributi:

- Numero

- Durata (in minuti)
- Titolo
- Numero di utenti (che hanno guardato l' episodio)

Opening. Una opening è identificata da un titolo, e contiene questi attributi:

- Titolo
- Compositore
- Durata

Persona. Una persona generica è identificata da un codice fiscale e contiene i seguenti attributi:

- Codice Fiscale
- Nome
- Data di nascita

Una persona si suddivide in **Utente**, **Nome** e **Registi**

Utenti. Un utente, oltre agli attributi di *Persona* contiene:

- Username
- Email

Attore. Un attore, oltre agli attributi di *Persona*, contiene:

- Nazionalita'
- Il numero di film in cui ha partecipato

Regista. Un regista, oltre agli attributi di *Persona*, contiene:

- Nazionalita
- Il genere di film creati

Piattaforma streaming. Una piattaforma streaming e' identificata dal suo nome e contiene i seguenti attributi:

- Nome
- Costo mensile

3 Progettazione concettuale

3.1 Lista entità

Il database si compone delle seguenti entità:

- **SerieTV:** Rappresenta una Serie Tv univoca, con una o più stagioni.
 - Titolo: string
 - AnnoInizio (ovvero l'anno della prima stagione): int
 - Genere: string
 - Descrizione: string
- **Stagione:** Rappresenta una stagione di una specifica Serie Tv
 - Numero_stagione: int
 - Anno: int
- **Episodio:** Rappresenta un singolo episodio di una specifica stagione
 - Numero_episodio: int
 - Durata (in minuti): int
 - Titolo: string

- Numero_utenti (che hanno guardato l'episodio): int
- **Opening:** Rappresenta la sigla iniziale di una SerieTv, se presente.
 - Titolo: string
 - Durata: int
 - Compositore: string
- **Persona:** Rappresenta una persona generica.
 - Codice Fiscale: int
 - Nome: string
 - Data_nascita: date
- **Utente:** Rappresenta una specializzazione di Persona, rappresenta un abbonato a una o più piattaforme streaming
 - Username: string
 - Email: string
- **Attore:** Rappresenta una specializzazione di Persona, partecipa a una o più serie TV.
 - Nazionalità: string
 - Numero_serie (in cui ha partecipato): int
- **Regista:** Rappresenta una specializzazione di Persona, ha diretto una o più serie TV.
 - Nazionalita: string
 - Genere_serie: string
- **Piattaforma streaming:** Rappresenta una piattaforma che contiene una o più serie Tv, e a cui gli utenti possono abbonarsi con vari tipi di abbonamento.
 - Nome: string
 - Costo_mensile: float

3.2 Lista relazioni

- Serie TV $\Rightarrow (1, N) \Rightarrow$ ComposizioneStag $\Leftarrow (1,1) \Leftarrow$ Stagione
 - Una Serie Tv è composta da una o più stagioni
 - Una stagione compone una sola Serie Tv
- Stagione $\Rightarrow (1, N) \Rightarrow$ ComposizioneEP $\Leftarrow (1,1) \Leftarrow$ Episodio
 - Una stagione è composta da uno o più episodi
 - Un episodio compone una sola stagione
- Serie TV $\Rightarrow (1,1) \Rightarrow$ Contenimento $\Leftarrow (1, N) \Leftarrow$ Piattaforma-Streaming

- Una Serie Tv è contenuta (può essere vista) in una sola piattaforma streaming
- Una piattaforma streaming può contenere una o più serie tv
- Piattaforma-Streaming $\Rightarrow (1, N) \Rightarrow$ Sottoscrizione $\Leftarrow (1, N) \Leftarrow$ Utente
 - Ad una piattaforma streaming si sottoscrivono uno o più utenti
 - Un utente può sottoscrivere a una o più piattaforme streaming (minimo una per essere considerato utente)
 - Questa relazione contiene i seguenti attributi:
 - * Data_inizio: date
 - * Data_fine: date
 - * Tpo_abbonamento: string

- $Utente \Rightarrow (0, N) \Rightarrow Visiona \Leftarrow (0, N) \Leftarrow Episodio$
 - Un utente può vedere 0 o più episodi di stagioni differenti
 - Un episodio generico può essere visto da nessuno o da più utenti
 - Questa relazione contiene i seguenti attributi:
 - * Data: date
 - * Voto: int

Il voto è opzionale.
- $Serie\ TV \Rightarrow (1, N) \Rightarrow Performance \Leftarrow (1, N) \Leftarrow Attore$
 - In una serie tv recitano uno o più attori
 - Un attore può recitare in una o più serie tv
 - Questa relazione contiene i seguenti attributi:
 - * Ruolo: string
 - * Compenso: int
- $Serie\ TV \Rightarrow (1, 1) \Rightarrow Direzione \Leftarrow (1, N) \Leftarrow Regista$
 - Una serie tv è diretta da un solo regista
 - Un regista può dirigere una o più serie
- $Serie\ Tv \Rightarrow (0, 1) \Rightarrow ContenimentoOpening \Leftarrow (1, 1) \Leftarrow Opening$
 - Una serie tv contiene zero o una opening
 - Una opening appartiene a una sola Serie Tv

3.3 Lista generalizzazioni

Persona è una generalizzazione totale di **UTENTE**, **ATTORE** e **REGISTA**

3.4 Schema E-R

In **figura 1** è riportato il diagramma E-R che riassume i requisiti discussi in precedenza.

4 Progettazione Logica

In questa sezione illustriamo la traduzione dello schema concettuale in schema logico, per rappresentare i dati nel modo più efficace possibile. Andiamo quindi ad analizzare le ridondanze, per comprendere se sia preferibile eliminarle o mantenerle. Procediamo poi con l'eliminazione delle generalizzazioni e infine mostriamo lo schema ristrutturato.

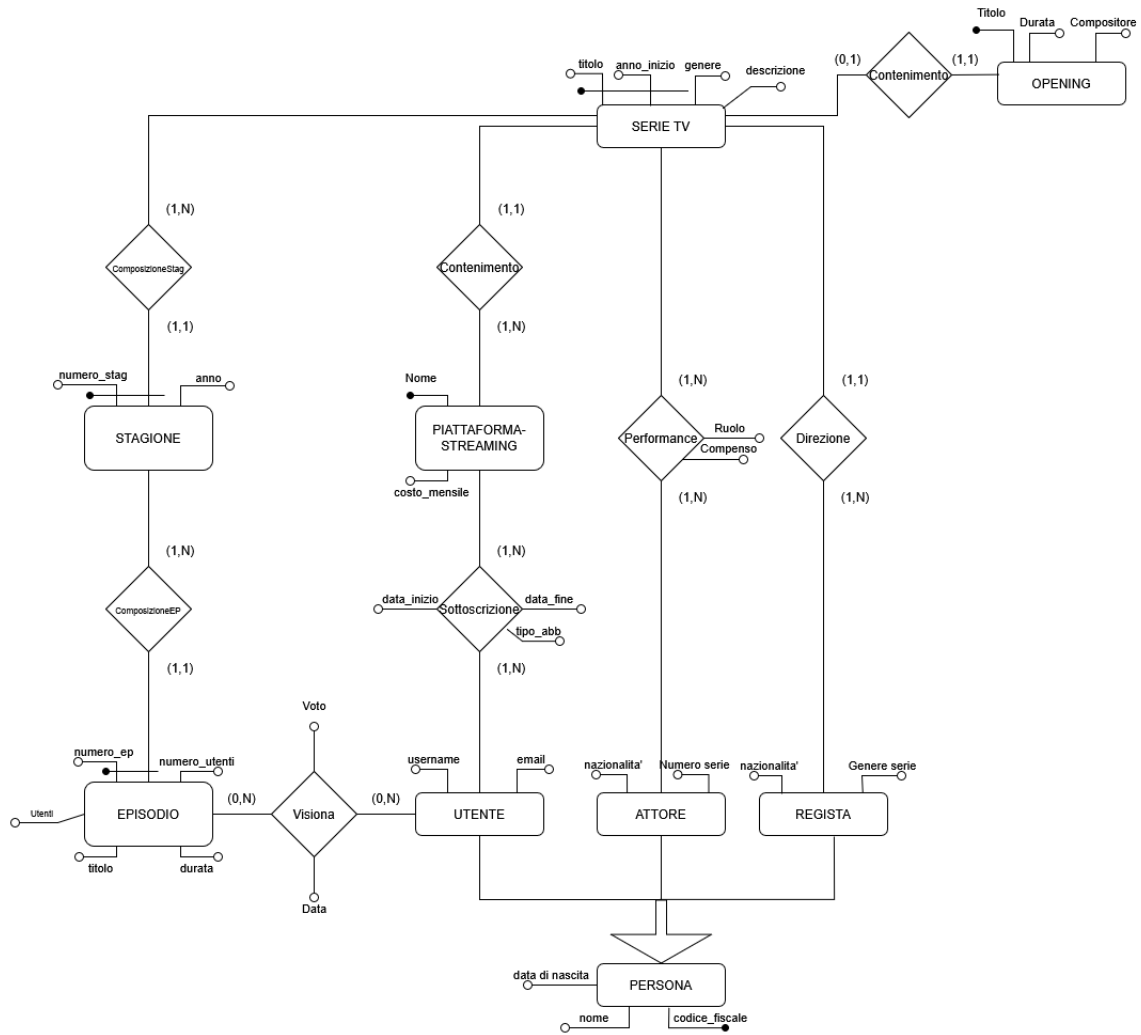


Figura 1: Schema E-R non ristrutturato

4.1 Analisi delle ridondanze

4.1.1 Ridondanza "NumeroSerieTv"

Possiamo notare che attributo “Numero Serie Tv” del sottoinsieme “Attore” entità “Persona” può essere calcolato dalla relazione Performance, poiché è il conteggio di tutte le Serie Tv per quel determinato attore. Questo attributo viene modificato (almeno una volta) ogni volta che viene inserita una nuova SerieTv (stimiamo circa 5 nuovi inserimenti al giorno) e viene visualizzato ogni volta un utente vuole sapere in quanti film ha recitato un attore (stimiamo 25 al giorno). Il tutto si riassume nelle seguenti operazioni:

- **Operazione 1 (5 al giorno):** inserimento di una nuova tupla in Performance
- **Operazione 2 (25 al giorno):** Visualizzare in quanti film ha recitato un attore.

Assumiamo i seguenti dati:

Concetto	Costrutto	Volume
Attore	Entità	1000
Performance	Relazione	2000

Con Ridondanza

- Operazione 1

Concetto	Costrutto	Accessi	Tipo	
Serie TV	E	1	S	$\times 5$
Performance	R	1	S	$\times 5$
Attore	E	1	L	$\times 5$
Attore	E	1	S	$\times 5$

Operazione 2

Concetto	Costrutto	Accessi	Tipo	
Attore	E	1	L	$\times 25$

Assumiamo costo doppio per le operazioni in scrittura, il costo totale ottenuto è: $((5 \times 3) \times 2) + 5 + 25 = 60$

Senza Ridondanza

- Operazione 1

Concetto	Costrutto	Accessi	Tipo	
Serie Tv	E	1	S	$\times 5$
Performance	R	1	S	$\times 5$

- Operazione 2

Stimiamo circa 2.000 performance attoriali

Concetto	Costrutto	Accessi	Tipo	
Attore	E	1	L	$\times 25$
Performance	R	2.000	L	$\times 25$

Assumiamo costo doppio per le operazioni in scrittura, il costo totale ottenuto è: $((5 \times 2) \times 2) + 25 + (2000 \times 25) \times 2 = 100.045$

Questo calcolo permette di stabilire che mantenere la ridondanza è preferibile, in modo da ottimizzare gli accessi.

4.1.2 Ridondanza "NumeroUtenti"

L'attributo "Numero utenti" dell'entità Episodio (relativa al numero di utenti che hanno guardato l'episodio) può essere calcolato a partire dalla relazione "Visione", in quanto è il conteggio di tutti gli utenti che hanno guardato quell'episodio. L'attributo viene modificato ogni volta che un utente guarda un episodio e viene visualizzato 50 volte al giorno per controllare l'andamento della piattaforma. Il tutto si riassume nelle seguenti operazioni:

- **Operazione 1 (10.000 al giorno):** inserimento di una nuova tupla in Visiona
- **Operazione 2 (50 al giorno):** Visualizzare il numero di utenti che hanno guardato un episodio.

Assumiamo i seguenti dati:

Concetto	Costrutto	Volume
Episodio	Entità	10.000
Visiona	Relazione	1.000.000

Con ridondanza

- Operazione 1

Concetto	Costrutto	Accessi	Tipo	
Visione	R	1	S	$\times 10.000$
Episodio	E	1	L	$\times 10.000$
Episodio	E	1	S	$\times 10.000$

- Operazione 2

Concetto	Costrutto	Accessi	Tipo	
Utenti	E	1	L	$\times 25$

Senza ridondanza

- Operazione 1

Concetto	Costrutto	Accessi	Tipo	
Visione	R	1	S	$\times 10.000$

Assumiamo costo doppio per le operazioni in scrittura, il costo totale ottenuto è: $(10.000 \times 2) \times 2 + 10.000 + 50 = 50.050$

- Operazione 2

Concetto	Costrutto	Accessi	Tipo	
Episodio	E	1	L	$\times 200$
Performance	R	1.000.000	L	$\times 200$

Assumiamo costo doppio per le operazioni in scrittura, il costo totale ottenuto è: $(10.000 \times 2) + 200 + (200 \times 2) = 20.600$

In questo caso invece è preferibile eliminare la ridondanza.

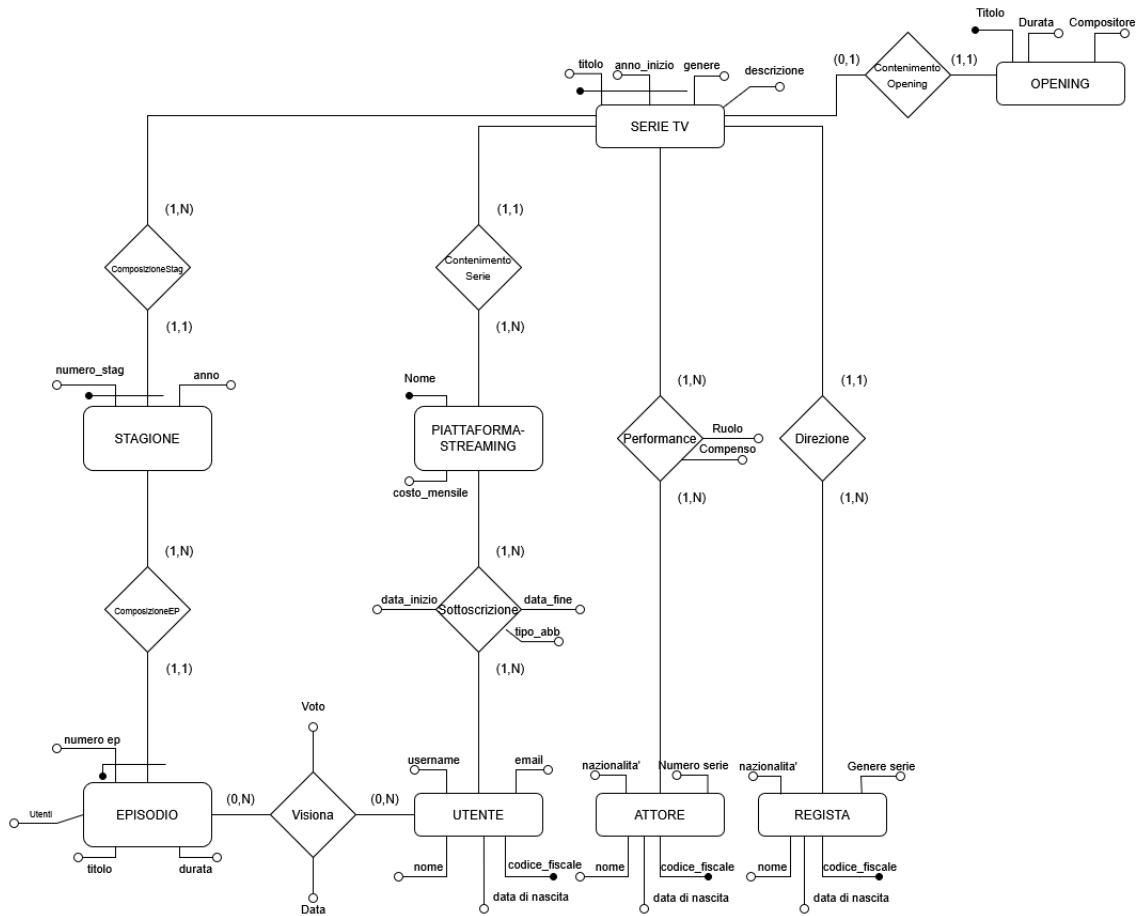


Figura 2: Schema E-R ristrutturato

4.2 Eliminazione delle generalizzazioni

Persona è una generalizzazione totale di **UTENTE**, **ATTORE** e **REGISTA**, presenta tre attributi *Nome*, *Data di nascita* e *Identificatore*, ma non è legata a nessuna relazione. Si nota quindi come questa generalizzazione possa essere eliminata. È preferibile infatti accorpare i suoi attributi nelle entità figlie. In questo modo si riduce il numero di attributi NULL rispetto all'operazione inversa, ovvero accorpare gli attributi delle entità figlie in una sola entità padre.

Inoltre, è possibile eliminare l'attributo "codice_fiscale" nell'entità "UTENTE" in quanto possiamo utilizzare l'attributo "username" come chiave primaria. Inoltre conoscere il codice fiscale di un utente può essere considerato sostanzialmente insensato e rischioso per la sicurezza.

In figura 2 è riportato il diagramma E-R ristrutturato, in cui sono state effettuate le modifiche discusse precedentemente.

4.3 Schema relazionale

In seguito è riportato lo schema logico costruito a partire dallo schema E-R ristrutturato. L'asterisco indica gli attributi che possono essere NULL.

- **Serie**(Titolo, AnnoInizio, Genere, Descrizione, Regista, PiattaformaStreaming, Opening)

- Serie.Regista=Regista.CodiceFiscale
- Serie.PiattaformaStreaming=PiattaformaStreaming.Nome
- **Stagione**(TitoloSerie, AnnoSerie, NumeroStagione, Anno)
 - Stagione.TitoloSerie=Serie.Titolo
 - Stagione.AnnoSerie=Serie.AnnoInizio
- **Episodio**(TitoloSerie, AnnoSerie, NumeroStagione, NumeroEpisodio, Durata, Titolo)
 - Episodio.TitoloSerie=Stagione.TitoloSerie
 - Episodio.AnnoSerie=Stagione.AnnoSerie
 - Episodio.NumeroStagione=Stagione.NumeroStagione
- **Opening**(TitoloSerie, AnnoSerie, Titolo, Compositore, Durata)
- **Visualizzazione**(Username, TitoloSerie, AnnoSerie, NumeroStagione, NumeroEpisodio, Data, Voto*)
 - Visualizzazione.Username=Utente.Username
 - Visualizzazione.TitoloSerie=Episodio.TitoloSerie
 - Visualizzazione.AnnoSerie=Episodio.AnnoSerie
 - Visualizzazione.NumeroStagione=Episodio.NumeroStagione
 - Visualizzazione.NumeroEpisodio=Episodio.Numero
- **Utente**(Username, Nome, DataNascita, Email)
- **Attore**(CodiceFiscale, Nome, DataNascita, Nazionalita, NumeroSerie)
- **Regista**(CodiceFiscale, Nome, DataNascita, Nazionalita, GenereSerie)
- **PiattaformaStreaming**(Nome, CostoMensile)
- **Sottoscrizione**(Username, Piattaforma, DataInizio, DataFine*, TipoAbbonamento)
 - Sottoscrizione.Username=Utente.Username
 - Sottoscrizione.Piattaforma=PiattaformaStreaming.Nome
- **Performance**(IDAttore, TitoloSerie, AnnoSerie, Ruolo, Compenso)
 - Performance.Attore=Attore.ID
 - Performance.TitoloSerie=Serie.Titolo
 - Performance.AnnoSerie=Serie.AnnoInizio

5 Implementazione in PostgreSQL e Definizione delle Query

Definizione delle Query

Di seguito vengono presentate e descritte le query con i relativi output generati e viene motivato l'utilizzo dell'indice proposto.

Query 1

Obiettivo: Trovare tutti i titoli delle serie TV disponibili su una piattaforma specifica, ad esempio *Netflix*.

```
SELECT Titolo
FROM Serie
WHERE PiattaformaStreaming = 'Netflix';

SELECT stv.titolo, COUNT(*) as numero_stagioni, op.titolo
FROM serie_tv stv
JOIN stagione st
ON st.titolo_serie=stv.titolo AND st.anno_serie=stv.anno_inizio
JOIN opening op
ON op.titolo_serie=stv.titolo AND op.anno_serie=stv.anno_inizio
WHERE piattaforma_streaming= 'Netflix'
GROUP BY stv.titolo, op.titolo
```

Listing 1: Esempio di query SQL

Query 2

Obiettivo: Calcolare la media dei voti per ogni episodio della serie *Dark*.

```

SELECT E.Titolo, AVG(V.Voto) AS MediaVoto
FROM Episodio E
JOIN Visualizzazione V ON
    E.TitoloSerie = V.TitoloSerie AND
    E.AnnoSerie = V.AnnoSerie AND
    E.NumeroStagione = V.NumeroStagione AND
    E.NumeroEpisodio = V.NumeroEpisodio
WHERE E.TitoloSerie = 'Dark'
GROUP BY E.Titolo;

SELECT E.TITOLO_episodio, AVG(V.Voto) as MediaVoto
FROM Episodio E
JOIN VISUALIZZAZIONE V ON
    E.Titolo_Serie = V.Titolo_Serie AND
    E.Anno_Serie = V.Anno_Serie AND
    E.Numero_Stagione = V.Numero_Stagione AND
    E.Numero_Episodio = V.Numero_Episodio
WHERE E.Titolo_Serie = 'Help-desk multi-laterale globale'
GROUP BY E.Titolo_episodio
HAVING AVG(V.Voto)>7.5

```

Listing 2: Esempio di query SQL

Query 3

Obiettivo: Numero di visualizzazioni totali per piattaforma

```

SELECT V.Username, COUNT(*) AS EpisodiVisti
FROM Visualizzazione V
GROUP BY V.Username
HAVING COUNT(*) >= 5;

--DA CONTROLLARE--
SELECT
    ps.nome AS piattaforma,
    COALESCE(vstats.totale_visualizzazioni, 0) AS totale_visualizzazioni,
    COALESCE(sstats.utenti_iscritti, 0) AS utenti_iscritti
FROM

```

```

    PIATTAFORMA_STREAMING ps

-- subquery per visualizzazioni per piattaforma
LEFT JOIN (
    SELECT
        st.piattaforma_streaming AS piattaforma,
        COUNT(*) AS totale_visualizzazioni
    FROM
        VISUALIZZAZIONE v
    JOIN
        SERIE_TV st ON v.titolo_serie = st.titolo AND v.anno_serie = st.
                    anno_inizio
    GROUP BY
        st.piattaforma_streaming
) AS vstats ON ps.nome = vstats.piattaforma

-- subquery per numero utenti iscritti per piattaforma
LEFT JOIN (
    SELECT
        nome_piattaforma AS piattaforma,
        COUNT(DISTINCT username) AS utenti_iscritti
    FROM
        SOTTOSCRIZIONE
    GROUP BY
        nome_piattaforma
) AS sstats ON ps.nome = sstats.piattaforma

ORDER BY
    totale_visualizzazioni DESC;

```

Listing 3: Esempio di query SQL

Query 4

Obiettivo: Trovare gli attori che hanno recitato in almeno una serie vista dall'utente `mario_rossi`.


```

SELECT DISTINCT A.Nome
FROM Performance P
JOIN Attore A ON P.IDAttore = A.ID
WHERE EXISTS (
    SELECT 1
    FROM Visualizzazione V
    WHERE V.Username = 'mario_rossi'
        AND V.TitoloSerie = P.TitoloSerie
        AND V.AnnoSerie = P.AnnoSerie
);

```

Query 5

Obiettivo: Trovare le prime 3 serie TV (per numero totale di visualizzazioni) dirette da registi italiani e disponibili su piattaforme con costo mensile superiore a 10 euro.

```

SELECT S.Titolo, COUNT(*) AS TotaleVisualizzazioni
FROM Serie S
JOIN Visualizzazione V ON
    S.Titolo = V.TitoloSerie AND
    S.AnnoInizio = V.AnnoSerie
JOIN Regista R ON S.Regista = R.ID
JOIN PiattaformaStreaming P ON S.PiattaformaStreaming = P.Nome
WHERE R.Nazionalita = 'Italiana' AND P.CostoMensile > 10
GROUP BY S.Titolo
ORDER BY TotaleVisualizzazioni DESC
LIMIT 3;

```