



# Progetto Basi di dati - YouTube

Creato da Picello Davide e Mahdi Mouad

## Abstract

YouTube è una piattaforma di condivisione video online che permette agli utenti di interagire con una vasta gamma di contenuti video. Sul sito è possibile vedere videoclip, trailer, cortometraggi, notizie, live streaming e altri contenuti come video blog, brevi video originali, video didattici e altro ancora, classificabili anche per età, con gli utenti che possono anche interagire votando, commentando, aggiungendo ai preferiti e, nei casi ove necessario, segnalando i video.

La maggioranza dei suoi video sono liberi per visualizzazione, ma ci sono eccezioni, inclusi i video caricati in modalità "premium" basati su un abbonamento solitamente mensile noleggiato film, nonché YouTube Premium.

## Analisi dei requisiti

Il progetto di YouTube prevede una base di dati che raccoglie informazioni sui **video(o live)** con dettagli come titolo, descrizione, data di pubblicazione, numero di visualizzazioni, **likes** e dislikes. Ogni video è associato a un canale, che rappresenta l'**account** dell'utente che ha caricato il video.



Google

@Google • 11 Mln di iscritti

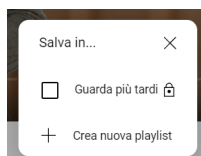
Experience the world of Google on our official YouTube channel. Watch videos about our products, technology, company ...

Esempio di un canale youtube

Gli **utenti** possono interagire con i video attraverso azioni come il like, il dislike, i commenti e l'aggiunta ai preferiti. Il progetto prevede anche una tabella per gestire le informazioni sui like e i dislike ricevuti da ciascun video, consentendo di calcolare il rating complessivo di un video basato sulla somma di queste valutazioni.



Bottoni di like e dislike



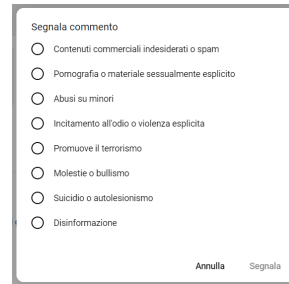
Finestra per salvare un video in un playlist

Inoltre, il progetto include la gestione delle **playlist**, che consentono agli utenti di creare elenchi personalizzati di video. Ogni playlist è associata a un account utente e può contenere una serie di video.

Per tenere traccia delle visualizzazioni dei video, è stata creata una tabella **"Views"** che registra l'account che ha

visualizzato il video, l'ID del video e la data della visualizzazione.

Oltre alle interazioni degli utenti, il progetto prevede anche la gestione delle **segnalazioni** dei video e dei commenti.



Finestra per segnalare un commento

Sono presenti due tabelle separate per registrare le segnalazioni dei video e dei commenti, con informazioni come l'account che ha effettuato la segnalazione, l'ID del video o del commento segnalato, il motivo della segnalazione e la data della segnalazione.

Infine, sono presenti le tabelle per gestire i **like** ai video e ai commenti. Ogni like è associato a un account utente, all'ID del video o del commento e alla data in cui è stato effettuato. Queste informazioni consentono di calcolare il rating complessivo di un video o di un commento basato sulla somma dei like e dei dislike.

In sintesi, il progetto di database per YouTube mira a gestire le informazioni relative ai video, agli utenti e alle interazioni tra questi fornendo delle statistiche su di esse.

## Glossario dei termini

Termine	Descrizione	Collegamenti
Account	Rappresenta un utente registrato su YouTube con informazioni come nome, email e data di registrazione. Un utente produce e/o interagisce con i contenuti della piattaforma	
Video	Rappresenta un video o una live caricato su YouTube, con attributi come il titolo, la descrizione, la durata e la data di pubblicazione.	Account
Views	Registra le visualizzazioni di un video, con informazioni sull'account che ha guardato il video, l'ID del video e la data di visualizzazione.	Account, Video
Abbonamenti	Rappresenta la relazione di abbonamento tra due canali su YouTube. Registra gli abbonamenti degli utenti a determinati canali, con informazioni sull'account che effettua l'abbonamento e il canale a cui si è abbonato.	Account(canale), Account(iscritto)
Playlist	Rappresenta una raccolta di video organizzati da un utente, con attributi come il titolo, la descrizione e l'account creatore della playlist.	Account, Video

Termine	Descrizione	Collegamenti
Commenti	Rappresenta i commenti lasciati dagli utenti su un video, con attributi come il messaggio del commento, l'account che ha commentato, l'ID del video e la data del commento.	Account, Video, Commenti(in caso di risposta ad un altro commento)
SavedPlaylist	Associa gli account agli ID delle playlist salvate, permettendo agli utenti di salvare playlist di loro interesse.	Account, Playlist
SegnalazioniCommenti	Registra le segnalazioni fatte dagli utenti su un commento, con informazioni sull'account che ha effettuato la segnalazione, l'ID del commento segnalato e il motivo della segnalazione.	Account, Commenti
SegnalazioniVideo	Registra le segnalazioni fatte dagli utenti su un video, con informazioni sull'account che ha effettuato la segnalazione, l'ID del video segnalato e il motivo della segnalazione.	Account, Video
LikeVideo	Registra i like e i dislike dati dagli utenti a un video, con informazioni sull'account che ha espresso il like/dislike, l'ID del video e la data dell'azione.	Account, Video
LikeCommenti	Registra i like e i dislike dati dagli utenti a un commento, con informazioni sull'account che ha espresso il like/dislike, l'ID del commento e la data dell'azione.	Account, Commenti

## Operazioni tipiche

- Operazioni tipiche di un server YouTube in una giornata (*stime approssimative: le frequenze effettive possono variare in base a molti fattori*)

Caricamento e processamento di nuovi video	Migliaia o addirittura milioni di volte al giorno, a seconda della quantità di contenuti caricati dagli utenti.
Visualizzazioni e conteggio di like/dislike	Milioni o miliardi di volte al giorno (poiché i video vengono visualizzati da un vasto numero di utenti).
Gestione degli abbonamenti e delle notifiche	Migliaia o milioni di volte al giorno
Elaborazione delle segnalazioni e delle richieste di rimozione dei contenuti	Migliaia o decine di migliaia di volte al giorno
Calcolo delle statistiche di visualizzazione e interazione:	Costantemente in tempo reale, mentre gli utenti interagiscono con i video.
Fornitura di suggerimenti e raccomandazioni personalizzate	Continuamente in tempo reale.
Monitoraggio e mitigazione degli abusi, spam e attività fraudolente	Costantemente in tempo reale, poiché YouTube deve individuare e affrontare rapidamente tali

	comportamenti indesiderati.
Archiviazione e gestione dei dati	Costantemente in tempo reale
Manutenzione del sistema e risoluzione dei problemi tecnici	Periodicamente o in risposta a problemi o necessità specifiche del sistema.

## Progettazione Concettuale

### Lista entità

#### Account

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Account	SERIAL	Codice numerico univoco per ogni account
handle	varchar(256) NOT NULL UNIQUE	Nome unico per ogni utente, modificabile dall'utente
mail	varchar(256) NOT NULL UNIQUE	Indirizzo email dell'account
password	varchar(256) NOT NULL	Password dell'account
dataIscrizione	date NOT NULL	Data di iscrizione dell'account
imgProfilo	varchar(256)	Link all'immagine del profilo
nome_Utente	varchar(256) NOT NULL	Nome dell'utente
cognome_Utente	varchar(256) NOT NULL	Cognome dell'utente
compleanno	timestamp	Data di compleanno dell'utente
genere	Gender NOT NULL	Genere dell'utente
paese	varchar(256) NOT NULL	Paese dell'utente
StatoAccount	Stato NOT NULL	Stato dell'account: descrive se l'account è sospeso, attivo o eliminato(quando si elimina un account da interfaccia utente non viene del tutto eliminato dal database)
descrizione	varchar(1000)	Descrizione dell'account, scritta dall'utente
premium	boolean NOT NULL	Indica se attivato l'abbonamento a YouTube premium

Vincoli:

- PRIMARY KEY: id\_Account

#### Video

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Video	SERIAL	ID unico per ogni video
titolo	varchar(256) NOT NULL	Titolo del video
descrizione	varchar(500)	Descrizione del video
dataPubblicazione	timestamp NOT NULL	Data di pubblicazione del video
durata	INT NOT NULL	Durata del video

Nome del campo	Tipo di dato + vincoli	Descrizione
costo	float	Costo del video (0 se gratuito)
categoria	Categorie	Categoria del video
visibilita	Visibilita DEFAULT 'Pubblico'	Visibilità del video
stato	Stato NOT NULL	Stato del video
id_Account	INT NOT NULL	ID dell'account che ha caricato il video
isLive	boolean	Flag che indica se il video è una live in corso (0 → no, 1 → sì)
dataFine	timestamp	Nel caso fosse una live, questo è il momento in cui finisce
thumbnail	varchar(256)	Link all'immagine di anteprima del video

Vincoli:

- PRIMARY KEY: id\_Video
- FOREIGN KEY: id\_Account REFERENCES Account(id\_Account) ON UPDATE CASCADE ON DELETE CASCADE
- CHECK: (dataFine IS NULL AND isLive = false) OR (dataFine IS NOT NULL AND isLive = true)

## Abbonamenti

Nome del campo	Tipo di dato + vincoli	Descrizione
canale	INT	ID del canale a cui ci si abbona
iscritto	INT	ID del canale di colui che si abbona
livello	Abbonamento NOT NULL	Livello di abbonamento. Se abbonamento = 'Gratis', solo iscrizione
dataIscrizione	timestamp NOT NULL	Data di iscrizione all'abbonamento

Vincoli:

- PRIMARY KEY: (canale, iscritto)
- FOREIGN KEY: canale REFERENCES Account(id\_Account) ON UPDATE CASCADE ON DELETE CASCADE
- FOREIGN KEY: iscritto REFERENCES Account(id\_Account) ON UPDATE CASCADE ON DELETE CASCADE
- CHECK: (iscritto <> canale) -- Controlla che uno non si iscriva a se stesso

## Views

Nome del campo	Tipo di dato + vincoli	Descrizione
account	INT NOT NULL	ID dell'account che sta guardando il video
id_Video	INT NOT NULL	ID del video che viene visualizzato

Nome del campo	Tipo di dato + vincoli	Descrizione
dataView	timestamp NOT NULL	Data e ora in cui è avvenuta la visualizzazione

Vincoli:

- PRIMARY KEY: (account, id\_Video)
- FOREIGN KEY: account REFERENCES Account(id\_Account) ON DELETE CASCADE ON UPDATE CASCADE
- FOREIGN KEY: id\_Video REFERENCES Video(id\_Video) ON DELETE CASCADE ON UPDATE CASCADE

## Playlist

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Playlist	SERIAL	ID univoco della playlist
account	INT NOT NULL	ID dell'account che ha creato la playlist
titolo	varchar(256) NOT NULL DEFAULT 'Guarda più tardi'	Titolo della playlist
descrizione	varchar(500)	Descrizione della playlist
visibilità	Visibilita DEFAULT 'Privato'	Visibilità della playlist, di default impostata a 'Privato'

Vincoli:

- PRIMARY KEY: id\_Playlist
- UNIQUE: (titolo, account)
- FOREIGN KEY: account REFERENCES Account(id\_Account) ON DELETE CASCADE ON UPDATE CASCADE

## VideoPlaylist

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Video	INT NOT NULL	ID del video salvato
id_Playlist	INT NOT NULL	ID della playlist in cui si salva il video

- PRIMARY KEY: (id\_Video, id\_Playlist),
- FOREIGN KEY(id\_Video) REFERENCES Video(id\_Video) ON DELETE CASCADE ON UPDATE CASCADE,
- FOREIGN KEY(id\_Playlist) REFERENCES Playlist(id\_Playlist) ON DELETE CASCADE ON UPDATE CASCADE

## Savedplaylist

Nome del campo	Tipo di dato + vincoli	Descrizione
account	INT NOT NULL	Account che salva la playlist

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Playlist	INT NOT NULL	ID della playlist salvata

La definizione della tabella include i seguenti vincoli:

- PRIMARY KEY: (account, id\_Playlist)
- FOREIGN KEY: account REFERENCES Account(id\_Account) ON DELETE CASCADE ON UPDATE CASCADE
- FOREIGN KEY: id\_Playlist REFERENCES Playlist(id\_Playlist) ON DELETE CASCADE ON UPDATE CASCADE

## Commenti

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Commento	SERIAL	Codice univoco per ogni commento
account	INT NOT NULL	Chi commenta
id_Video	INT NOT NULL	Che video commenta
messaggio	varchar(500) NOT NULL	Testo del commento
donazione	float	Campo per eventuali donazioni
dataCommento	timestamp NOT NULL	Data e ora del commento
id_Risposta	INT	Eventuale ID di un altro commento al quale si risponde

La definizione della tabella include i seguenti vincoli:

- PRIMARY KEY: id\_Commento
- FOREIGN KEY: account REFERENCES Account(id\_Account) ON DELETE CASCADE ON UPDATE CASCADE
- FOREIGN KEY: id\_Video REFERENCES Video(id\_Video) ON DELETE CASCADE ON UPDATE CASCADE
- FOREIGN KEY: id\_Risposta REFERENCES Commenti(id\_Commento) ON DELETE CASCADE ON UPDATE CASCADE
- CHECK: donazione >= 0

## SegnalazioniVideo

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Segnalazione	SERIAL	Codice univoco per ogni segnalazione
account	INT NOT NULL	Chi segnala
id_Video	INT NOT NULL	Video segnalato
motivo	Motivo	Motivo della segnalazione
descrizione	varchar(50)	Descrizione della segnalazione
data	timestamp NOT NULL	Data di segnalazione

Le chiavi primarie e le chiavi esterne sono mantenute come nell'originale:

- PRIMARY KEY: id\_Segnalazione

- FOREIGN KEY: account REFERENCES (id\_Account) ON DELETE CASCADE ON UPDATE CASCADE
- FOREIGN KEY: id\_Video REFERENCES (id\_Video) ON DELETE CASCADE ON UPDATE CASCADE

## SegnalazioniCommenti

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Segnalazione	SERIAL	Codice univoco per ogni segnalazione
account	INT NOT NULL	Chi segnala
id_Commento	INT NOT NULL	Commento segnalato
motivo	Motivo	Motivo della segnalazione
data	timestamp NOT NULL	Data di segnalazione

Le chiavi primarie e le chiavi esterne sono mantenute come nell'originale:

- PRIMARY KEY: id\_Segnalazione
- FOREIGN KEY: account REFERENCES Account(id\_Account) ON DELETE CASCADE ON UPDATE CASCADE
- FOREIGN KEY: id\_Commento REFERENCES Commenti(id\_Commento) ON DELETE CASCADE ON UPDATE CASCADE

## LikeVideo

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Like	SERIAL	Codice univoco per ogni like ai video
account	INT NOT NULL	Account che ha messo il like/dislike
id_Video	INT NOT NULL	Video a cui è stato messo il like/dislike
data	timestamp NOT NULL	Momento dell'assegnazione del like/dislike
valuation	Likes NOT NULL	Valutazione del like/dislike (-1 = dislike, 1 = like)

- PRIMARY KEY: id\_Like
- FOREIGN KEY: account REFERENCES Account(id\_Account) ON UPDATE CASCADE ON DELETE CASCADE
- FOREIGN KEY: id\_Video REFERENCES Video(id\_Video) ON UPDATE CASCADE ON DELETE CASCADE

## LikeCommenti

Nome del campo	Tipo di dato + vincoli	Descrizione
id_Like	SERIAL	Codice univoco per ogni like ai commenti



Nome del campo	Tipo di dato + vincoli	Descrizione
account	INT NOT NULL	Account che ha messo il like/dislike
id_Commento	INT NOT NULL	Commento a cui è stato messo il like/dislike
data	timestamp NOT NULL	Momento dell'assegnazione del like/dislike
valuation	Likes NOT NULL	Valutazione del like/dislike (-1 = dislike, 1 = like)

- PRIMARY KEY: id\_Like
- FOREIGN KEY: account REFERENCES Account(id\_Account) ON UPDATE CASCADE ON DELETE CASCADE
- FOREIGN KEY: id\_Commento REFERENCES Commenti(id\_Commento) ON UPDATE CASCADE ON DELETE CASCADE

## Lista delle Relazioni

Account-Video:

- un video appartiene ad account (1,1)
- un account ha diversi video (1,N)

Account-Playlist

- una playlist è creata da un solo account (1,1)
- un account ha una o più playlists associate(almeno 2 "mi piace" e "guarda più tardi") (2,N)

Views-Video

- un video ha diverse views (0,N)
- una views appartiene ad un ed un solo video (1,1)

Account-Views

- un account ha diverse views associate ad esso (0,N)
- una views appartiene ad un unico canale (1,1)

Account-Commenti

- un account ha diversi commenti associate ad esso (0,N)
- un commento appartiene ad un unico canale (1,1)

Account-SavedPlaylist

- un account almeno 2 playlist salvate( "mi piace" e "guarda più tardi") (2:N)
- una playlist è salvata da diverse persone (1,N)

Account-Abbonamento

- una account ha diversi abbonati (0,N)
- un account è abbonato a diversi canali (0,N)

Account-LikeCommento

- un account può mettere like o dislike a commenti diversi(0,N)

- un like ad un commento appartiene ad un solo account (1,1)

#### Account-LikeVideo

- un account può mettere like o dislike a video diversi (0,N)
- un like ad un commento appartiene ad un solo account (1,1)

#### Playlist-SavedPlaylist

- una playlist può essere salvata da diversi utenti (1,N)
- un utente può avere diverse playlist salvate(almeno “mi piace” e “guarda più tardi” che gli appartengono) (2:N) -

#### Video-LikeVideo

- un video ha diversi associati ad esso (0,N)
- un like è associato ad un solo video (1,1)

#### Commento-LikeCommento

- un commento può avere diversi like (0,N)
- un like a commento è associato ad un solo commento (1,1)

#### Video-SegnalazioniVideo

- un video può ricevere diverse segnalazioni (0,N)
- una segnalazione si riferisce ad un solo video (1,1)

#### Commenti-SegnalazioniCommenti

- un commento può avere diverse segnalazioni (0,N)
- una segnalazione appartiene ad un unico commento (1,1)

#### Playlist-Video

- un video può appartenere a diverse playlists (0,N)
- una playlist ha diversi video (0,N)

#### Account-SegnalazioniCommenti

- un account può fare diverse segnalazioni a commenti (0,N)
- una segnalazione a commento è fatta da un solo account (1,1)

#### Account-SegnalazioniVideo

- un account può fare diverse segnalazioni a video (0,N)
- una segnalazione a video è fatta da un solo account (1,1)

#### VideoPlaylist-Video

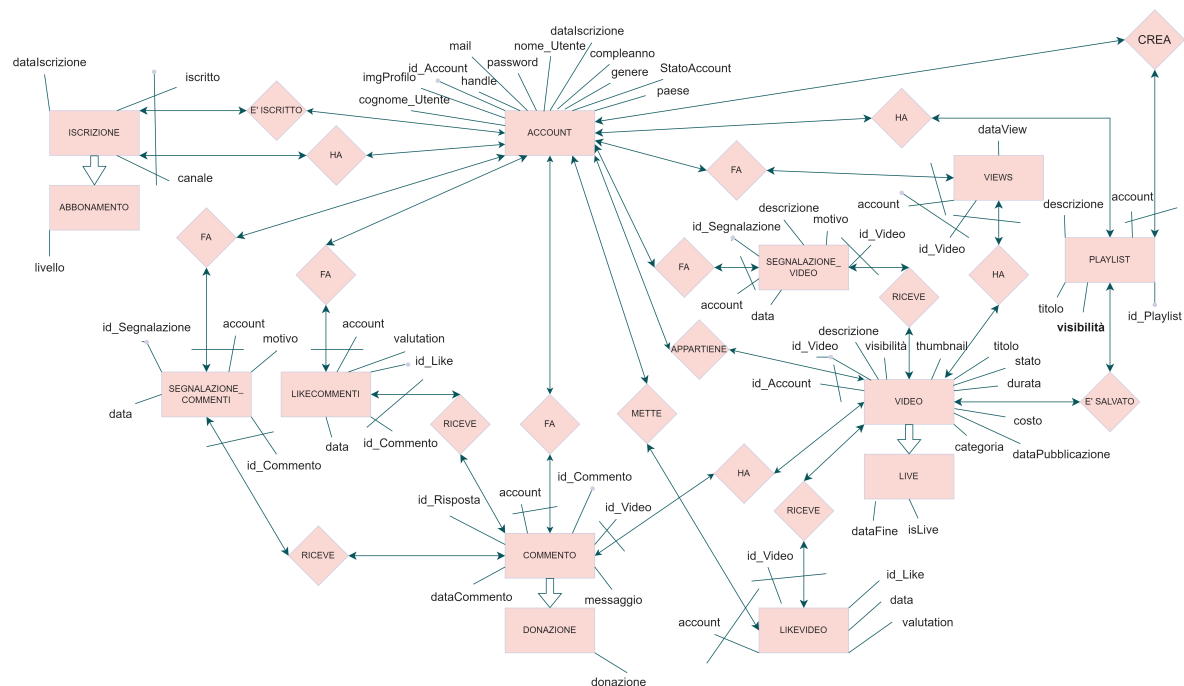
- un video può essere salvato in diverse playlist (0:N)
- una playlist può avere diversi video (1:N)

#### VideoPlaylist-Playlist

- Una playlist può essere composta da 1 o più video (1:N)
- Un video può essere salvato da 0 o più playlist (0:N)

# Analisi logica

## Schema concettuale



## Analisi e gestione delle ridondanze ed ereditarietà

### Ridondanze

Nelle fasi iniziali della progettazione, questo progetto, aveva un problema di ridondanza.

Non esistevano infatti le tabelle **VideoPlaylist** e **SavedPlaylist**. Il che comportava che **Playlist** avesse come attributi: **id\_Playlist**, **account**, **id\_Video**, **titolo**, **descrizione**, **visibilità** e **account** di chi la salvava.

Questo comportava gravi problemi di ridondanza in quanto, per ogni video salvato, avremmo salvato, in modo ridondante, gli attributi: **account**, **titolo**, **descrizione**, **visibilità** → tutti attributi inerenti alla playlist e non ai video salvati.

Per risolvere questo problema, parte della tabella **Playlist** (la parte relativa ai video) è stata salvata in una nuova tabella, **VideoPlaylist** e **SavedPlaylist**, che hanno il compito di mantenere le informazioni dei video salvati e di chi le salva, eliminando così la ridondanza.

**VideoPlaylist** e **SavedPlaylist** hanno il minimo indispensabile per poter sapere quale video è salvato in quale playlist e chi ha salvato quella playlist.

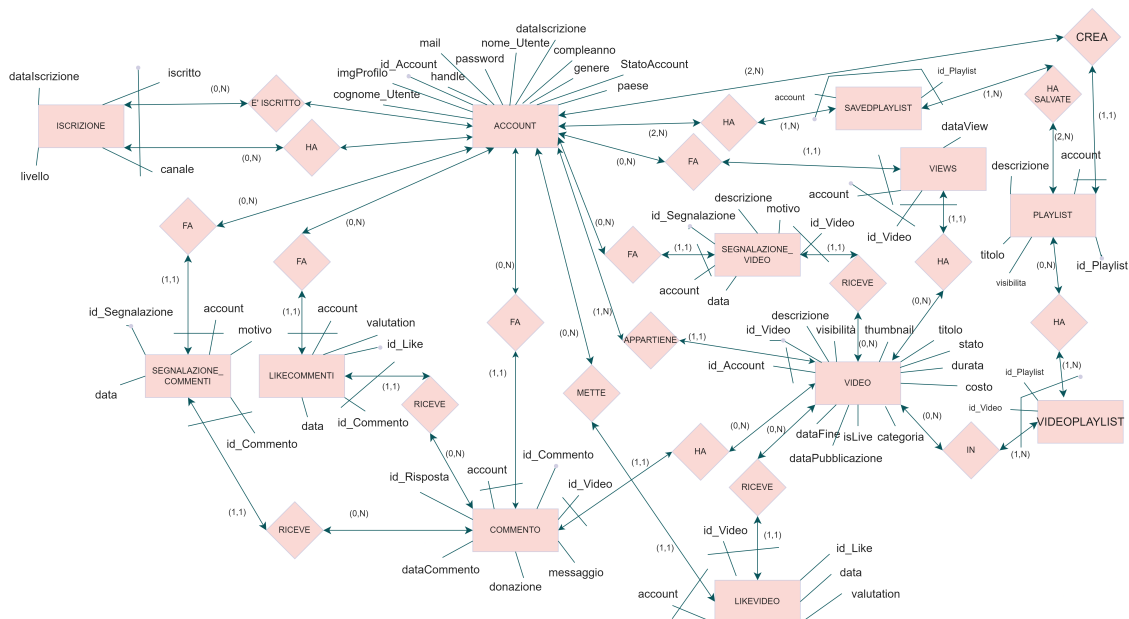
### Ereditarietà (eliminazione delle generalizzazioni)

Le generalizzazioni sono state gestite incorporandole nelle classi principali.

Nello specifico:

- **Iscrizione** incorpora **abbonamento** includendo "livello" e impostandolo a 0 se non si è abbonati
- **Commento** incorpora **donazione** includendo "donazione" equivalente a 0 se non avviene.
- **Video** incorpora **live** includendo gli attributi "isLive" e "dataFine" rispettivamente impostati a 0 se la live è finita, o si tratta di un semplice video, ed a NULL se si tratta di un video: se invece si fosse trattato di una live terminata avremmo avuto la sua data di fine.

# Diagramma E-R finale



## Queries

### Queries SQL

1) Creazione view per video e stampa delle visualizzazioni per canale:

```
CREATE VIEW ViewsPerVideo AS(
SELECT id_Video, COUNT(account) AS total_views
FROM Views
GROUP BY id_Video
);

SELECT Video.id_account, SUM(total_views) AS ViewsCanale
FROM ViewsPerVideo, Video
GROUP BY Video.id_account
HAVING SUM(total_views) > 10;
```

2) Query che crea le playlist guarda più tardi e video piaciuti per ogni utente:

```
INSERT INTO Playlist (account, titolo, descrizione, visibilita)
SELECT id_Account, 'Guarda più tardi', ' ', 'Privato'
FROM account;

INSERT INTO Playlist (account, titolo, descrizione, visibilita)
SELECT id_Account, 'Video piaciuti', 'descrizione', 'Privato'
FROM account;
```

3) Query che aggiunge video a cui ogni utente ha messo mi piace alla playlist video piaciuti:

```

INSERT INTO VideoPlaylist(id_Video,id_Playlist)
SELECT L.id_Video, P.id_Playlist
FROM Playlist AS P, LikeVideo AS L
WHERE L.valutation='1' AND L.account=P.account AND P.titolo='Video piaciuti'

```

#### 4) Video ordinati in base al rating ricevuto(like+dislike/views):

```

DROP VIEW IF EXISTS ViewsPerVideo;
DROP VIEW IF EXISTS SommaLike;
DROP VIEW IF EXISTS voto;

CREATE VIEW ViewsPerVideo AS(
SELECT id_Video, COUNT(account) AS total_views
FROM Views
GROUP BY id_Video);

CREATE VIEW voto AS(
SELECT id_Video,
CASE
WHEN valutation = '1' THEN 1.0
ELSE -1.0
END AS stato
FROM LikeVideo);

CREATE VIEW SommaLike AS(
SELECT id_Video, SUM(stato) AS SommaLike
FROM voto
GROUP BY id_Video);

SELECT S.id_Video, (S.SommaLike/V.total_views) AS RATING
FROM SommaLike AS S JOIN ViewsPerVideo AS V ON S.id_Video=V.id_Video
ORDER BY RATING DESC;

```

#### 5) Video in tendenza per views(top 10):

```

SELECT V.*
FROM Video AS V
JOIN (
SELECT id_Video, COUNT(*) AS num_views
FROM Views
GROUP BY id_Video
) AS VIEWS ON V.id_Video = VIEWS.id_Video
ORDER BY VIEWS.num_views DESC
LIMIT 10;

```

#### 6) Utenti a cui far vedere pubblicità:

```

SELECT *
FROM account
WHERE premium = FALSE;

```

#### 7) Lista delle live in corso:

```

SELECT *
FROM Video
WHERE isLive = true;

```

8) Proiezione dei commenti: in questo caso una conversazione di commenti che rispondono al commento 1682

```
SELECT *  
FROM Commenti  
WHERE id_commento = 1682 OR id_risposta = 1682  
ORDER BY datacommento;
```

## Queries parametriche in C++

Per l'implementazione delle query parametriche è stato sviluppato un software C++ che, tramite un piccolo menù, permette di scegliere una tra le seguenti queries.

Ognuna chiederà in input un parametro, tranne l'ultima che ne chiede due.

### 1. Ricerca video

Questa query simula la ricerca di un determinato video .

Selezionando questa opzione, verrà chiesto di inserire in input il nome di un video.

Se esiste, ed è pubblico, verranno stampate alcune informazioni sul video e sul proprietario.

### 2. Ricerca account

Questa query simula la ricerca di un determinato account.

Selezionando questa opzione, verrà richiesto di inserire in input l'handle di un account.

Se l'account esiste e il suo stato è "Attivo", saranno restituite diverse informazioni sull'account.

### 3. Ricerca video per categoria

Selezionando questa opzione, verrà richiesto di inserire in input una categoria.

Saranno restituiti i video pubblici appartenenti a quella categoria.

### 4. Ricerca account correlati ai video di una categoria

Selezionando questa opzione, verrà richiesto di inserire in input una categoria.

Saranno restituite varie informazioni sugli account attivi correlati ai video appartenenti a quella categoria (basta avere un video per quella determinata categoria per apparire tra i risultati).

### 5. Ricerca video per account

Questa query simula la ricerca di tutti i video appartenenti ad un determinato account

Selezionando questa opzione, verrà richiesto di inserire in input l'handle di un account.

Saranno restituite informazioni sui video pubblici dell'account corrispondente.

### 6. Ricerca video per account e categoria

Questa query simula la ricerca di tutti i video appartenenti ad una determinata categoria, pubblicati da un determinato account.

Selezionando questa opzione, verrà richiesto di inserire in input un handle di account e una categoria.

Verranno restituiti i video di quel account appartenenti a quella determinata categoria.

## Indici

```
CREATE INDEX idx_likevideo_voto ON LikeVideo (valuation);

CREATE INDEX idx_video_titolo ON Video (titolo);

CREATE INDEX idx_views_account ON Views (account);
```

**idx\_likevideo\_voto** sulla tabella **LikeVideo**: Questo indice migliora il calcolo del rating dei video. Indicizza la colonna **valuation** nella tabella **LikeVideo**, consentendo un accesso più rapido ai dati relativi alle valutazioni (like/dislike) dei video.

**idx\_video\_titolo** sulla tabella **Video**: Questo indice è utile per le query parametriche, come la barra di ricerca. Indicizza la colonna **titolo** nella tabella **Video**, consentendo una ricerca più veloce e ottimizzata dei video in base al titolo.

**idx\_views\_account** sulla tabella **Views**: Questo indice facilita la visualizzazione della cronologia e il calcolo delle visualizzazioni per video e canali. Indicizza la colonna **account** nella tabella **Views**, consentendo un accesso rapido alle visualizzazioni associate a un account specifico.

## Popolamento database

Per il popolamento del database si è usato, per le tabelle più piccole, ChatGpt: molto utile per generare dati di fantasia e sempre diversi ma con grandi limiti riguardo a quantità di risultati prodotti e mantenimento dei vari vincoli tra le varie tabelle (per esempio è capitato che generasse commenti con una data antecedente alla data di pubblicazione del video). Tutti gli errori di questo tipo sono dovuti essere verificati e sistemati manualmente, causando molte perdite di tempo.

Per le tabelle più grandi invece sono stati realizzati dei software C++ che le popolassero.

Questo metodo richiede sicuramente più sforzo e tempo per essere realizzato, ma garantisce dei risultati di altissima qualità ed affidabilità rispetto ai risultati forniti da ChatGpt.

Nella cartella "Popolamento", sono allegati questi file.