Progetto Basi di Dati

Nicolò Nava [SM3201248] – 2 Febbraio 2025

Sommario

1. Introduzione

1.1 Descrizione

Un'organizzazione desidera realizzare un servizio di streaming in cui gli utenti possano:

- Registrarsi e fruire di un catalogo di video (film, documentari, serie TV).
- Sottoscrivere piani di abbonamento.
- Memorizzare gli eventi di visione (per sapere chi ha visto cosa, quando e con quale metodo di pagamento o abbonamento).

L'obiettivo della base di dati è gestire:

- Il catalogo di video offerti (insieme a informazioni come il produttore, il distributore, l'anno di uscita, ecc.).
- Gli utenti e i loro metodi di pagamento o piani di abbonamento.
- Gli eventi di fruizione (ad es. quando un utente guarda un certo contenuto).
- Altre informazioni come: premi vinti da un video, enti che rilasciano tali premi, localizzazioni geografiche (città) relative alle aziende e agli utenti.

Inoltre, per motivi di marketing e promozione, si vogliono memorizzare i premi ottenuti dai video (ad esempio un "Best Streaming Award"). Ciascun premio è rilasciato da un ente specializzato (riviste, siti web, festival) che possiede un numero di fan, un tipo, ecc.

Funzionamento generale

1. Gli utenti hanno:
    - Username, Nome, Cognome, Data di nascita.
    - Una città di residenza (e un indirizzo).
    - Possono avere metodi di pagamento di vario tipo (carta di credito, oppure un portafoglio interno) oppure possono attivare piani di abbonamento con pagamento ricorrente.
2. I metodi di pagamento sono caratterizzati da un ID, da un tipo (carta, portafoglio, ecc.). In caso di carta di credito ci sono ulteriori informazioni (nome, numero carta), mentre nel caso di portafoglio interno è presente un ammontare di credito disponibile.
3. I piani di abbonamento hanno un ID e specificano un costo mensile, una durata e un eventuale numero di schermi disponibili.

4. I video nel catalogo hanno un titolo, un produttore, un distributore, una data di uscita, un prezzo pay-per-view (usato se l'utente non ha un abbonamento o se il contenuto richiede costi extra) e un numero di visualizzazioni (per motivi statistici).
5. Un produttore e un distributore sono entrambi specializzazioni di una generica entità "azienda". Un'azienda ha un nome, un numero di dipendenti e una sede in una certa città.
6. Quando un utente guarda un video, si vuole memorizzare la cosiddetta visione comprensiva della data e del "metodo" (se l'utente paga singolarmente quel video oppure l'ha incluso nell'abbonamento). Ciò può essere modellato come una relazione ternaria (Utente, Video, Metodo/Abbonamento).
7. I premi assegnati a un video hanno un nome, una categoria, un anno di conferimento e l'ente che lo rilascia. L'ente, a sua volta, ha un nome, un tipo (festival, rivista specializzata, sito web, ecc.) e un numero di fan.
8. È possibile che un video venga prodotto e distribuito dalla stessa azienda, così come un'azienda potrebbe essere sia produttrice che distributrice in contesti diversi (generalizzazione sovrapposta).

1.2 Elenco Operazioni

Di seguito alcune **operazioni** che l'applicazione desidera svolgere (con frequenza e tipo di accesso):

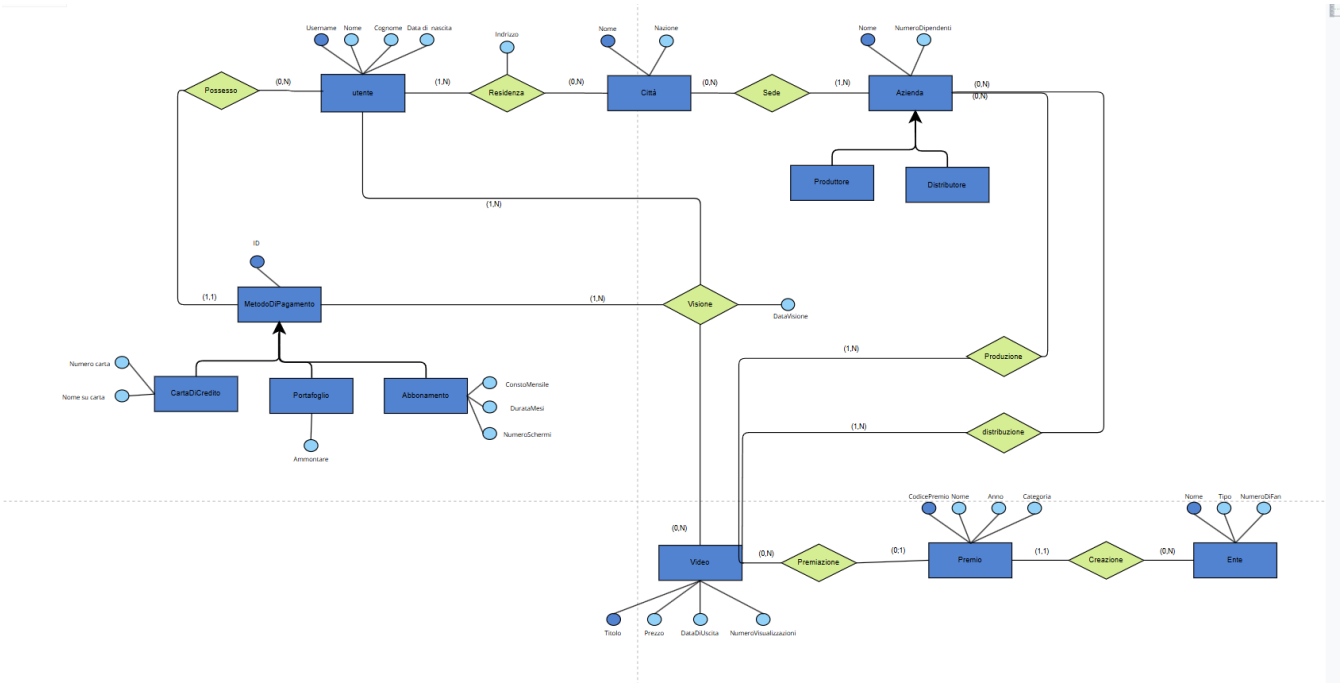
1. **(Interattiva, 2 volte al giorno)**
   L'applicazione desidera ottenere i titoli dei video visti da un certo utente, quanti ne ha in totale e con quale metodo di pagamento (o piano di abbonamento) siano stati fruiti, ordinati per data di fruizione.
2. **(Interattiva, 1 volta a settimana)**
   L'applicazione desidera ottenere tutte le informazioni dei video, incluso il numero di visualizzazioni, ordinati per numero di visualizzazioni (decrescente), insieme al numero di premi ottenuti da ogni video.
3. **(Interattiva, 3000 al giorno)**
   L'applicazione desidera memorizzare un nuovo evento di fruizione (visione) all'interno del database.
4. **(Interattiva, 2 volte al mese)**
   L'applicazione desidera ottenere le informazioni dei produttori e distributori (con almeno un certo numero di dipendenti) e le loro relative sedi, insieme al video più "costoso" (cioè con il prezzo pay-per-view più alto) presente nel catalogo.
5. **(Interattiva, 2 volte a settimana)**
   L'applicazione desidera ottenere, dato un premio (fornito come parametro), l'ente che

l'ha rilasciato, il video che l'ha ricevuto e le aziende che hanno prodotto e distribuito quel video.

6. **(Batch, 4 volte al giorno)**
   L'applicazione desidera aggiornare i dati relativi agli utenti (ad es. modifiche di nome, cognome, data di nascita o indirizzo).

2. Diagramma Entity-Relationship

2.1 Diagramma (versione concettuale)



Le entità principali sono:

- Utente
- MetodoDiPagamento (con generalizzazione a CartaDiCredito, Portafoglio, Abbonamento)
- Video
- Azienda (con generalizzazione a Produttore, Distributore)
- Premio
- Ente
- Città

Viene considerata la relazione ternaria Visione tra Utente, Video e MetodoDiPagamento, con l'attributo DataVisione. Sono inoltre presenti relazioni binarie quali Possesso (Utente–MetodoDiPagamento), Residenza (Utente–Città), Sede (Azienda–Città), Produzione (Produttore–Video), Distribuzione (Distributore–Video), Premiazione (Premio–Video), Creazione (Premio–Ente).

2.2 Considerazioni

La scelta di una relazione ternaria “Visione” risponde all’esigenza di legare in un solo costrutto gli oggetti Utente, Video e MetodoDiPagamento/Abbonamento.
La generalizzazione MetodoDiPagamento - (CartaDiCredito, Portafoglio, Abbonamento) è totale ed esclusiva. La generalizzazione Azienda - (Produttore, Distributore) è totale e sovrapposta (un’azienda può essere sia produttrice sia distributrice).

3. Osservazioni e tabelle

3.1 Vincoli Extra

- Ogni utente può possedere al più un Portafoglio tra i metodi di pagamento.
- Un’azienda deve essere almeno produttore o distributore, ma può svolgere entrambi i ruoli.

3.2 Dizionario dei dati per le entità

Entità	Descrizione	Attributi	Identificatore
Video	Contenuto multimediale del catalogo	Titolo, Prezzo, DataDiUscita, NumeroVisualizzazioni	Titolo
Utente	Utente registrato alla piattaforma	Username, Nome, Cognome, DataDiNascita	Username
MetodoDiPagamento	Metodo con cui l’utente effettua pagamenti o abbonamenti	ID, Tipo	ID
CartaDiCredito	Sottoentità di MetodoDiPagamento	(da MetodoDiPagamento) + Nome, NumeroCarta	ID (ereditato)
Portafoglio	Sottoentità di MetodoDiPagamento (credito interno)	(da MetodoDiPagamento) + Ammontare	ID (ereditato)
Abbonamento	Sottoentità di MetodoDiPagamento	(da MetodoDiPagamento) + CostoMensile, DurataMesi, NumSchermi	ID (ereditato)
Azienda	Azienda generica operante nel settore streaming	Nome, NumeroDipendenti	Nome
Produttore	Sottoentità di Azienda (specializzata in produzione)	(ereditati da Azienda)	Nome (ereditato)

Entità	Descrizione	Attributi	Identificatore
Distributore	Sottoentità di Azienda (specializzata in distribuzione)	(ereditati da Azienda)	Nome (ereditato)
Premio	Riconoscimento assegnato a un video	CodicePremio, Nome, Categoria, Anno	CodicePremio
Ente	Ente (rivista, sito, festival) che fornisce il premio	Nome, Tipo, NumeroDiFan	Nome
Città	Città di residenza o sede aziendale	Nome, Nazione	Nome

3.3 Dizionario dei dati per le relazioni

Relazione	Descrizione	Componenti	Attributi
Visione	Evento di fruizione: un utente guarda un video pagando/ricorrendo a un metodo o abbonamento	Utente, Video, MetodoDiPagamento	DataVisione
Possesso	Appartenenza di un metodo di pagamento a un utente	Utente, MetodoDiPagamento	–
Residenza	Città in cui risiede un utente	Utente, Città	Indirizzo
Sede	Città in cui è situata la sede di un'azienda	Azienda, Città	–
Produzione	Produzione di un video	Produttore (Azienda), Video	–
Distribuzione	Distribuzione di un video	Distributore (Azienda), Video	–
Premiazione	Assegnazione di un premio a un video	Premio, Video	–
Creazione	Attività di creazione di un premio da parte di un ente	Premio, Ente	–

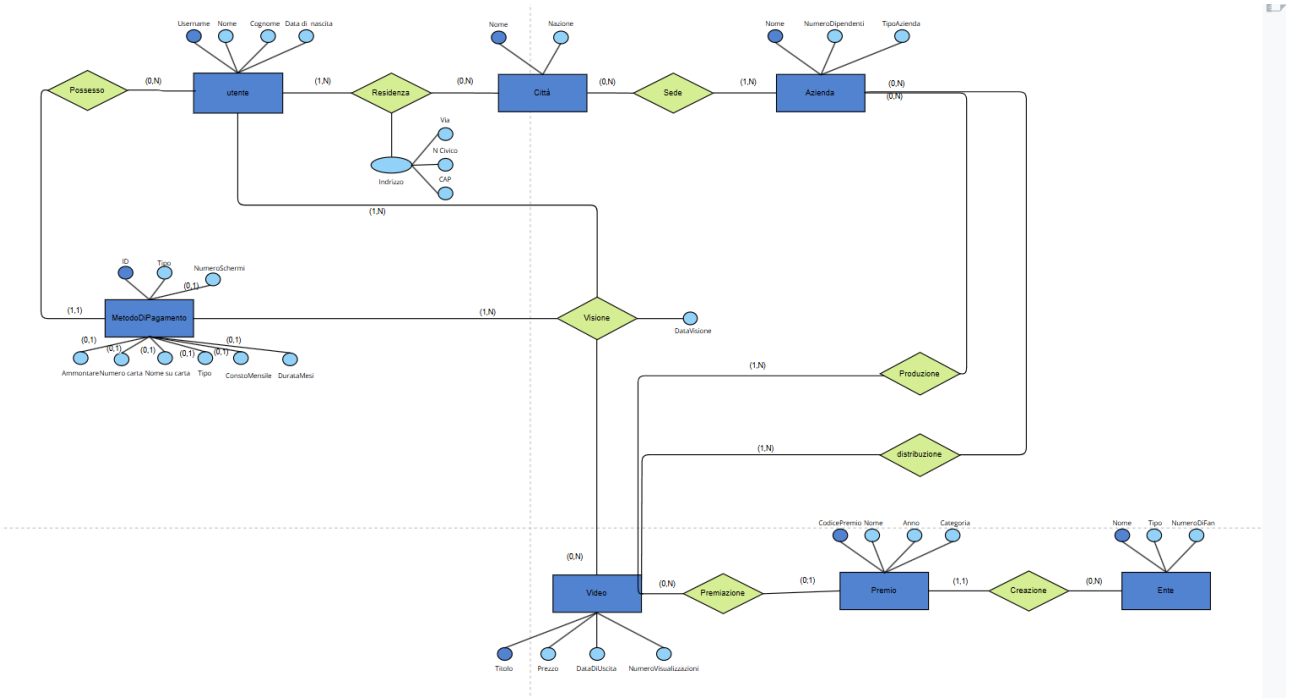
3.4 Volumi

Concetto	Tipo	Volume
Video	E	20.000

Concetto	Tipo	Volume
Utente	E	300.000
MetodoDiPagamento	E	350.000
CartaDiCredito	E	200.000
Portafoglio	E	50.000
Abbonamento	E	100.000
Azienda	E	2.000
Produttore	E	1.500
Distributore	E	1.000
Premio	E	1.000
Ente	E	100
Città	E	5.000
Visione	R	2.000.000
Possesso	R	350.000
Residenza	R	300.000
Sede	R	2.000
Produzione	R	20.000
Distribuzione	R	20.000
Premiazione	R	2.000
Creazione	R	1.000

4. Diagramma Entity-Relationship Ristrutturato

4.1 Diagramma



4.1 Scelta delle PK

- Viene definito un CodicePremio per l'entità Premio, al fine di ottenere un identificatore semplice.
- L'entità Video utilizza "Titolo".
- L'entità Utente utilizza "Username".
- L'entità MetodoDiPagamento utilizza "ID".
- L'entità Azienda utilizza "Nome".
- L'entità Ente utilizza "Nome".
- L'entità Città utilizza "Nome".

4.2 Analisi delle Ridondanze

È presente una possibile ridondanza con l'attributo NumeroVisualizzazioni in Video. Tale valore può essere ricavato con un conteggio delle occorrenze nella tabella delle visioni (Visione). Viene mantenuto l'attributo per motivi prestazionali.

4.3 Eliminazione delle Generalizzazioni

La generalizzazione relativa ai metodi di pagamento, se non serve distinguere (a livello di operazioni) i vari tipi, può essere rimossa e unificata in un'unica entità MetodoDiPagamento con attributi facoltativi (NomeCarta, NumeroCarta, Ammontare, CostoMensile, ecc.).
Nel caso di Azienda, la generalizzazione "Azienda - (Produttore, Distributore)" può essere eliminata a favore di un attributo "TipoAzienda" (Produttore, Distributore, Entrambi).

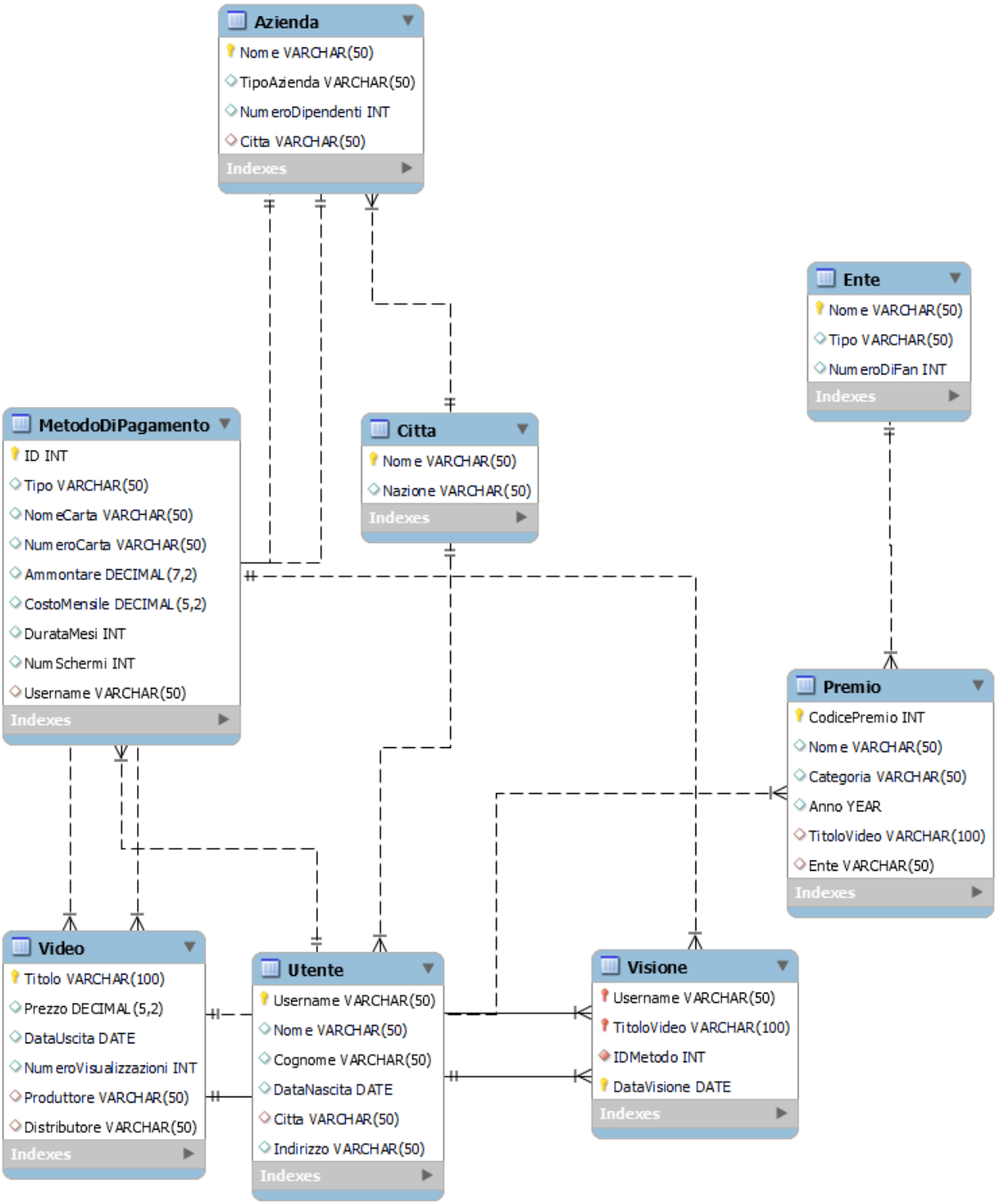
4.4 Accorpamento di Concetti

Gli attributi Via, Numero Civico, CAP possono essere accorpati in un singolo attributo "Indirizzo" associato alla relazione Residenza.

È possibile accorpare le sottoentità di MetodoDiPagamento (CartaDiCredito, Portafoglio, Abbonamento) in un'unica tabella con attributi specifici. Analogamente, può essere definito un attributo "TipoAzienda" all'interno della tabella Azienda per indicare se un'azienda è produttrice, distributrice o entrambe.

5. Schema Logico

5.1 Schema



Schema (ristrutturato e con ridondanza su NumeroVisualizzazioni):

1. Utente(
   Username PK,

Nome,
Cognome,
DataNascita,
Città,
Indirizzo
)

2. Città(
   Nome PK,
   Nazione
)

3. Azienda(
   Nome PK,
   TipoAzienda,
   NumeroDipendenti,
   Città
)

4. Video(
   Titolo PK,
   Prezzo DECIMAL(5,2),
   DataUscita DATE,
   NumeroVisualizzazioni INT,
   Produttore (FK su Azienda(Nome)),
   Distributore (FK su Azienda(Nome))
)

5. MetodoDiPagamento(
   ID PK,
   Tipo,
   NomeCarta,
   NumeroCarta,
   Ammontare DECIMAL(7,2),
   CostoMensile DECIMAL(5,2),
   DurataMesi INT,
   NumSchermi INT,
   Username (FK su Utente(Username))
)

6. Visione(
   Username (FK su Utente(Username)),
   TitoloVideo (FK su Video(Titolo)),
   IDMetodo (FK su MetodoDiPagamento(ID)),

DataVisione DATE,
PRIMARY KEY(Username, TitoloVideo, DataVisione)
)

7. Ente(
   Nome PK,
   Tipo,
   NumeroDiFan INT
)
8. Premio(
   CodicePremio PK,
   Nome,
   Categoria,
   Anno YEAR,
   TitoloVideo (FK su Video(Titolo)),
   Ente (FK su Ente(Nome))
)

5.2 Analisi delle normalizzazioni

- Tutte le tabelle sono in 1FN (attributi atomici).
- È rispettata la 2FN (nessun attributo dipende solo da una parte di chiave, laddove presente).
- La 3FN risulta non pienamente soddisfatta per la presenza dell'attributo NumeroVisualizzazioni in Video, che potrebbe essere derivato da Visione. La scelta di mantenerlo risponde a esigenze di performance.
- Nel caso in cui Ammontare di un Portafoglio debba dipendere strettamente da un utente (non solo da ID), sarebbero possibili ulteriori normalizzazioni, ma ciò comporterebbe tabelle aggiuntive. Anche in questo caso viene preferito un compromesso prestazionale.

6. Query e Implementazioni

6.1 User Defined Function (UDF)

Funzione che restituisce il numero di premi di un video (impiegabile nell'operazione che richiede di mostrare il numero di premi ottenuti):

```sql
CREATE FUNCTION dbo.UdfCountPremiVideo
(
    @titoloVideo VARCHAR(30)
)
RETURNS INT
AS
BEGIN
    DECLARE @count INT;
    SELECT @count = COUNT(*)
    FROM Premio
    WHERE TitoloVideo = @titoloVideo;
    RETURN @count;
END;
GO
```

Funzione che restituisce il tipo di un'azienda (Produttore, Distributore o Entrambi) data la chiave primaria:

```sql
CREATE FUNCTION dbo.UdfTipoAzienda
(
    @nomeAzienda VARCHAR(30)
)
RETURNS VARCHAR(20)
AS
BEGIN
    DECLARE @tipo VARCHAR(20);
    SELECT @tipo = TipoAzienda
    FROM Azienda
    WHERE Nome = @nomeAzienda;
    RETURN @tipo;
END;
GO
```

6.2 Viste e Stored Procedure

Vista che elenca i video e il numero di premi (impiegabile per mostrare i risultati dell'operazione 2):

```sql
CREATE VIEW dbo.VistaVideoPremi
AS
SELECT
    v.Titolo,
    v.Prezzo,
    v.NumeroVisualizzazioni,
    dbo.UdfCountPremiVideo(v.Titolo) AS NumeroPremi
FROM Video v;
GO
```

Stored Procedure per inserire un nuovo evento di visione (operazione 3), con aggiornamento della ridondanza:

```
CREATE PROCEDURE dbo.SpuInserisciVisione
(
    @username       VARCHAR(15),
    @titoloVideo    VARCHAR(30),
    @idMetodo       INT,
    @dataVisione    DATE
)
AS
BEGIN
    INSERT INTO Visione (Username, TitoloVideo, IDMetodo, DataVisione)
    VALUES (@username, @titoloVideo, @idMetodo, @dataVisione);

    -- Aggiorna il numero di visualizzazioni
    UPDATE Video
    SET NumeroVisualizzazioni = NumeroVisualizzazioni + 1
    WHERE Titolo = @titoloVideo;
END;
GO
```

6.3 Esempi di

6.3 Triggers:

Trigger per limitare a un singolo Portafoglio per utente:

```
CREATE TRIGGER trgLimitaPortafoglio
ON MetodoDiPagamento
FOR INSERT
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM inserted i
        WHERE i.Tipo = 'Portafoglio'
        GROUP BY i.Username
        HAVING COUNT(*) > 1
    )
    BEGIN
        RAISERROR('Esiste gia` un portafoglio per questo utente', 16, 1);
        ROLLBACK TRANSACTION;
    END
END;
GO
```

Trigger per validare l'attributo TipoAzienda nella tabella **Azienda**:

```
CREATE TRIGGER trgValidaTipoAzienda
ON Azienda
FOR INSERT, UPDATE
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM inserted
        WHERE TipoAzienda NOT IN ('Produttore', 'Distributore', 'Entrambi')
    )
    BEGIN
        RAISERROR('TipoAzienda non valido', 16, 1);
        ROLLBACK TRANSACTION;
    END
END;
GO
```

Trigger per controllare coerenza di un metodo di pagamento (ad esempio, se Tipo = 'Carta' allora NumeroCarta non deve essere NULL):

```
CREATE TRIGGER trgCheckMetodoDiPagamento
ON MetodoDiPagamento
FOR INSERT, UPDATE
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM inserted
        WHERE Tipo = 'Carta'
          AND (NumeroCarta IS NULL OR NomeCarta IS NULL)
    )
    BEGIN
        RAISERROR('Carta di credito non valida', 16, 1);
        ROLLBACK TRANSACTION;
    END
END;
GO
```

Trigger per forzare la coerenza tra "Produttore" e "Distributore" nella tabella **Video**, verificando che le aziende coinvolte abbiano effettivamente il tipo "Produttore", "Distributore" oppure "Entrambi" (generalizzazione sovrapposta).

```
ON Video
FOR INSERT, UPDATE
AS
BEGIN
    -- Verifica che il produttore specificato abbia tipoAzienda 'Produttore' o 'Entrambi'
    IF EXISTS (
        SELECT 1
        FROM inserted i
        JOIN Azienda a ON i.Produttore = a.Nome
        WHERE a.TipoAzienda NOT IN ('Produttore', 'Entrambi')
    )
    BEGIN
        RAISERROR('Il produttore indicato non risulta essere un produttore o azienda con ruolo "Entrambi".', 16, 1);
        ROLLBACK TRANSACTION;
    END;

    -- Verifica che il distributore specificato abbia tipoAzienda 'Distributore' o 'Entrambi'
    IF EXISTS (
        SELECT 1
        FROM inserted i
        JOIN Azienda a ON i.Distributore = a.Nome
        WHERE a.TipoAzienda NOT IN ('Distributore', 'Entrambi')
    )
    BEGIN
        RAISERROR('Il distributore indicato non risulta essere un distributore o azienda con ruolo "Entrambi".', 16, 1);
        ROLLBACK TRANSACTION;
    END;
END;
GO
```

Trigger per impedire l'inserimento o l'aggiornamento di una data di visione futura nella tabella **Visione**.

```
CREATE TRIGGER trgCheckDataVisione
ON Visione
FOR INSERT, UPDATE
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM inserted
        WHERE DataVisione > GETDATE()
    )
    BEGIN
        RAISERROR('La data di visione non può essere futura.', 16, 1);
        ROLLBACK TRANSACTION;
    END;
END;
GO
```