Virtual Idol Management Database

Team 2

Rina Su - 002308528
Krisha Lakhani - 002334794
Adwait Relekar - 002839566
Reetika Bhanushali - 002341895
Prayag Adhikari - 002657754

01

INTRODUCTION

Introducing Virtual YouTubers:
The Rise of Digital Personalities

What is a Virtual YouTuber (VTuber)?
Definition of VTuber:
-A VTuber is an **online content creator**
-Uses a computer-generated avatar instead of showing their real face.

Data Management:
VTuber profiles **Key features:**

Content (live streams, videos, and channels)
Audience interactions and engagement metrics
Business activities (sponsorships, merchandise, and partnerships)

Digital Persona:
The avatars are often inspired by anime-style characters.
Technology Used: Motion Capture and Virtual Tools

Project Purpose:
Our database is designed to manage information, content, audience interactions, and business data for VTubers. It aims to provide a unified platform that simplifies:
Tracking of VTuber information and **activities**
Management of **video content** and **live streams**
Analysis of social media interactions and **viewer engagement**
Monitoring of revenue streams and **partnerships**
Optimization of **content strategies** to support growth

02

BUSINESS RULES

Business **Rules**

Virtual Idol Relationships:

Each Virtual Idol:

- Has exactly **one YouTube channel**.
- Can perform **only one live stream** at a time.
- Can utilize **one or many platforms** for content distribution.
- Can use **zero or multiple vocal changers**.

Collaborations:

Each Virtual Idol works with:

- **One painter** for image creation.
- **One motion capture specialist** for animation and movement.

Audience Engagement:

Each viewer:

- Can follow **multiple YouTube channels**.
- May write **zero or many reviews** for content.
- Can participate in **zero or multiple donations** to support idols.

Live Stream Content:

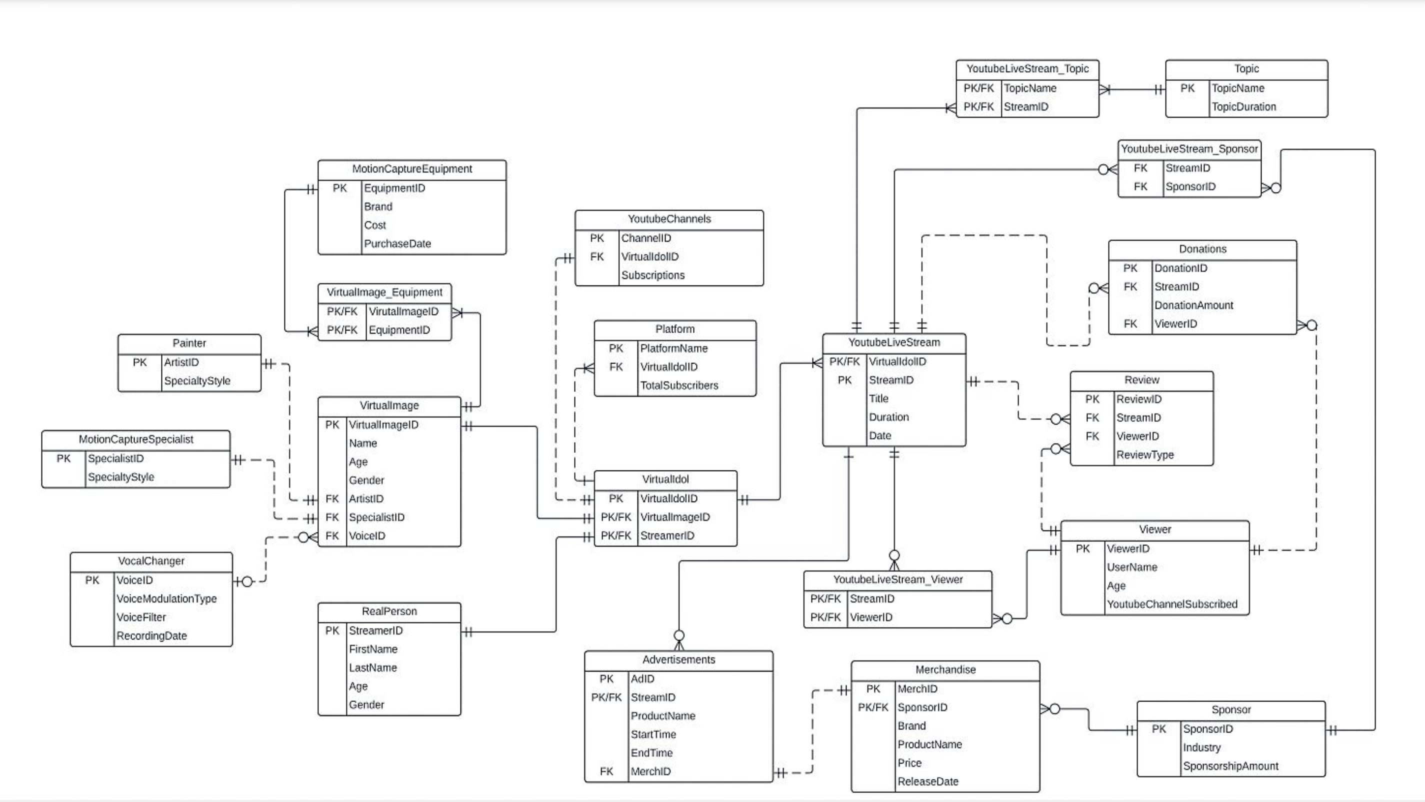
Each live stream:

- Can discuss **one or many topics**.

03

Entity Relationship Diagram

Entity Relationship Diagram(ERD)



	MotionCaptureEquipment
PK	EquipmentID
	Brand
	Cost
	PurchaseDate

	VirtualImage_Equipment
PK/FK	VirutalImageID
PK/FK	EquipmentID

	Painter
PK	ArtistID
	SpecialtyStyle

	MotionCaptureSpecialist
PK	SpecialistID
	SpecialtyStyle

	VocalChanger
PK	VoiceID
	VoiceModulationType
	VoiceFilter
	RecordingDate

	VirtualImage
PK	VirtualImageID
	Name
	Age
	Gender
FK	ArtistID
FK	SpecialistID
FK	VoiceID

	RealPerson
PK	StreamerID
	FirstName
	LastName
	Age
	Gender

	YoutubeChannels
PK	ChannelID
FK	VirtualIdolID
	Subscriptions

	Platform
PK	PlatformName
FK	VirtualIdolID
	TotalSubscribers

	VirtualIdol
PK	VirtualIdolID
PK/FK	VirtualImageID
PK/FK	StreamerID

	Advertisements
PK	AdID
PK/FK	StreamID
	ProductName
	StartTime
	EndTime
FK	MerchID

	YoutubeLiveStream_Topic
PK/FK	TopicName
PK/FK	StreamID

	Topic
PK	TopicName
	TopicDuration

	YoutubeLiveStream_Sponsor
FK	StreamID
FK	SponsorID

	Donations
PK	DonationID
FK	StreamID
	DonationAmount
FK	ViewerID

	YoutubeLiveStream
PK/FK	VirtualIdolID
PK	StreamID
	Title
	Duration
	Date

	Review
PK	ReviewID
FK	StreamID
FK	ViewerID
	ReviewType

	YoutubeLiveStream_Viewer
PK/FK	StreamID
PK/FK	ViewerID

	Viewer
PK	ViewerID
	UserName
	Age
	YoutubeChannelSubscribed

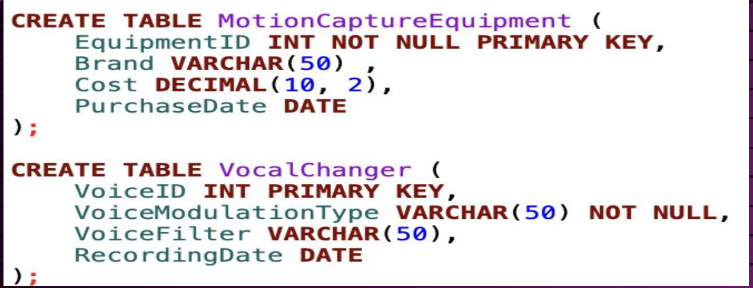
	Merchandise
PK	MerchID
PK/FK	SponsorID
	Brand
	ProductName
	Price
	ReleaseDate

	Sponsor
PK	SponsorID
	Industry
	SponsorshipAmount

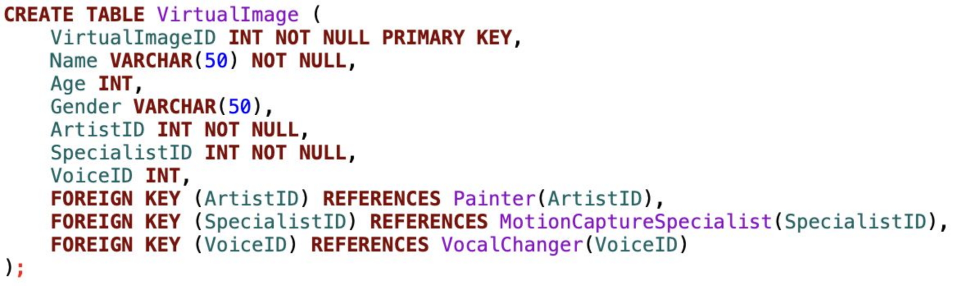
04

DATABASE IMPLEMENTATION

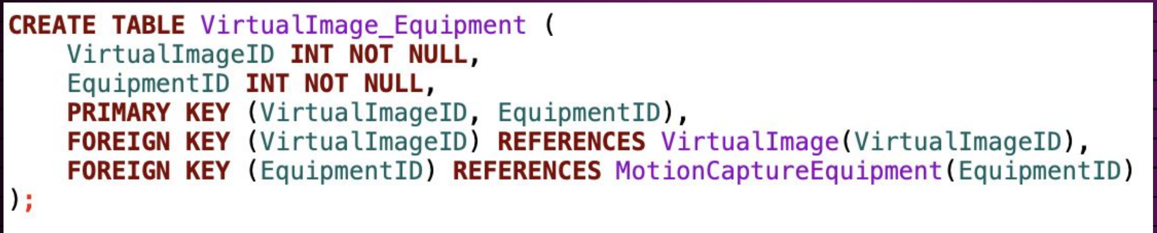
Table Creation



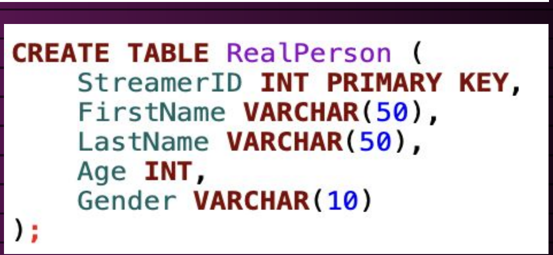
```
CREATE TABLE MotionCaptureEquipment (
    EquipmentID INT NOT NULL PRIMARY KEY,
    Brand VARCHAR(50) ,
    Cost DECIMAL(10, 2),
    PurchaseDate DATE
);

CREATE TABLE VocalChanger (
    VoiceID INT PRIMARY KEY,
    VoiceModulationType VARCHAR(50) NOT NULL,
    VoiceFilter VARCHAR(50),
    RecordingDate DATE
);
```



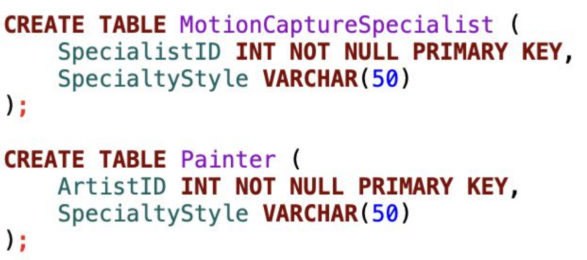
```
CREATE TABLE VirtualImage (
    VirtualImageID INT NOT NULL PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    Age INT,
    Gender VARCHAR(50),
    ArtistID INT NOT NULL,
    SpecialistID INT NOT NULL,
    VoiceID INT,
    FOREIGN KEY (ArtistID) REFERENCES Painter(ArtistID),
    FOREIGN KEY (SpecialistID) REFERENCES MotionCaptureSpecialist(SpecialistID),
    FOREIGN KEY (VoiceID) REFERENCES VocalChanger(VoiceID)
);
```



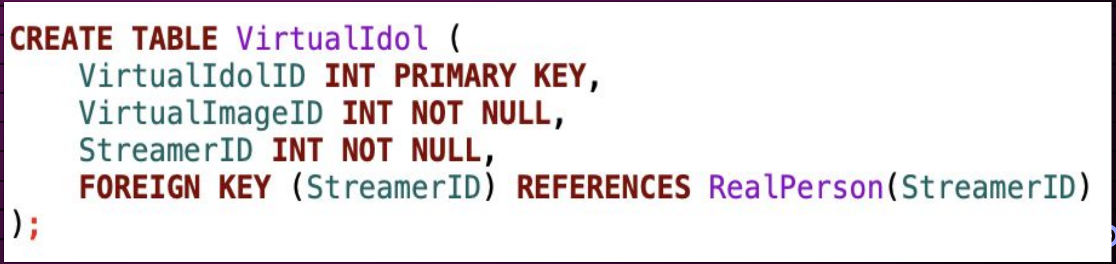
```
CREATE TABLE VirtualImage_Equipment (
    VirtualImageID INT NOT NULL,
    EquipmentID INT NOT NULL,
    PRIMARY KEY (VirtualImageID, EquipmentID),
    FOREIGN KEY (VirtualImageID) REFERENCES VirtualImage(VirtualImageID),
    FOREIGN KEY (EquipmentID) REFERENCES MotionCaptureEquipment(EquipmentID)
);
```



```
CREATE TABLE RealPerson (
    StreamerID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Age INT,
    Gender VARCHAR(10)
);
```

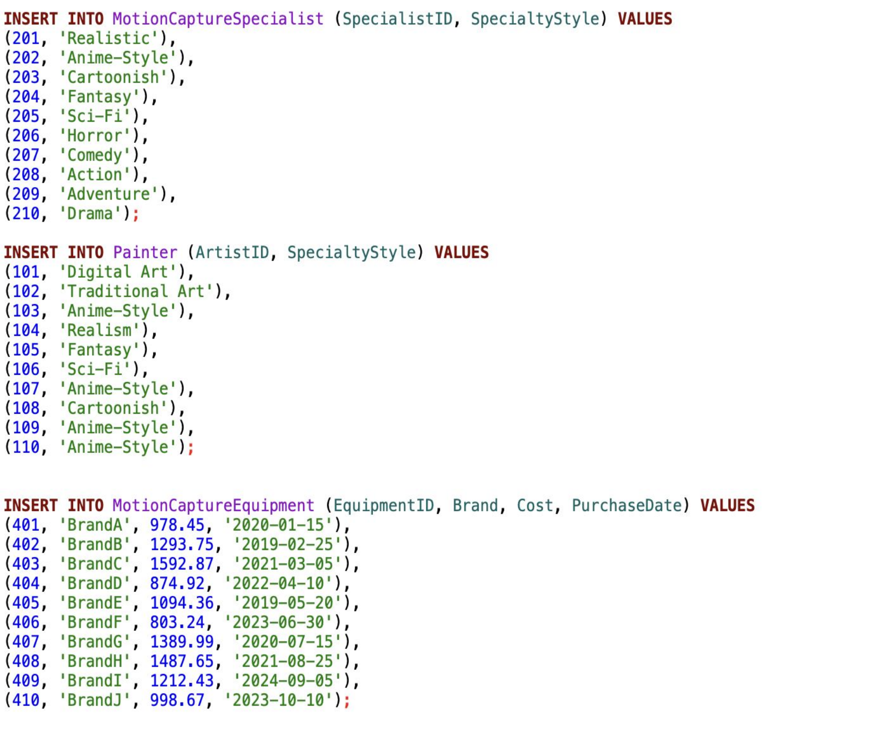


```
CREATE TABLE MotionCaptureSpecialist (
    SpecialistID INT NOT NULL PRIMARY KEY,
    SpecialtyStyle VARCHAR(50)
);

CREATE TABLE Painter (
    ArtistID INT NOT NULL PRIMARY KEY,
    SpecialtyStyle VARCHAR(50)
);
```

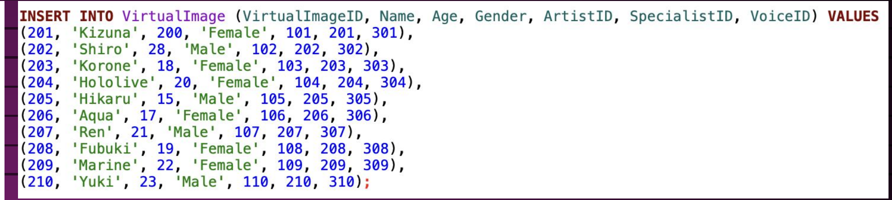


```
CREATE TABLE VirtualIdol (
    VirtualIdolID INT PRIMARY KEY,
    VirtualImageID INT NOT NULL,
    StreamerID INT NOT NULL,
    FOREIGN KEY (StreamerID) REFERENCES RealPerson(StreamerID)
);
```

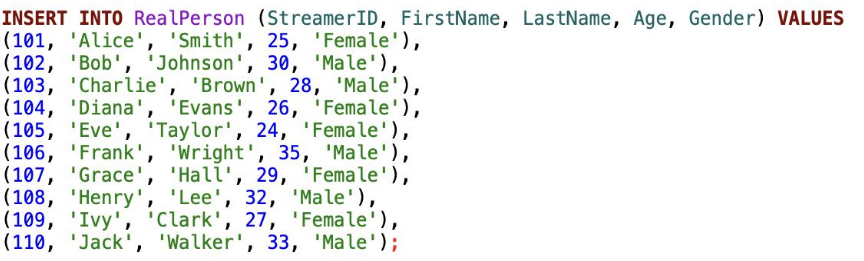
Data Insertion



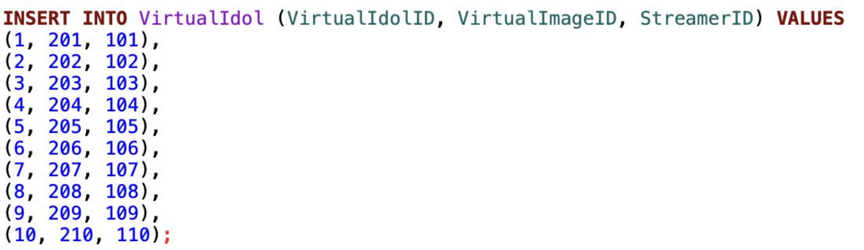
```sql
INSERT INTO MotionCaptureSpecialist (SpecialistID, SpecialtyStyle) VALUES
(201, 'Realistic'),
(202, 'Anime-Style'),
(203, 'Cartoonish'),
(204, 'Fantasy'),
(205, 'Sci-Fi'),
(206, 'Horror'),
(207, 'Comedy'),
(208, 'Action'),
(209, 'Adventure'),
(210, 'Drama');

INSERT INTO Painter (ArtistID, SpecialtyStyle) VALUES
(101, 'Digital Art'),
(102, 'Traditional Art'),
(103, 'Anime-Style'),
(104, 'Realism'),
(105, 'Fantasy'),
(106, 'Sci-Fi'),
(107, 'Anime-Style'),
(108, 'Cartoonish'),
(109, 'Anime-Style'),
(110, 'Anime-Style');


INSERT INTO MotionCaptureEquipment (EquipmentID, Brand, Cost, PurchaseDate) VALUES
(401, 'BrandA', 978.45, '2020-01-15'),
(402, 'BrandB', 1293.75, '2019-02-25'),
(403, 'BrandC', 1592.87, '2021-03-05'),
(404, 'BrandD', 874.92, '2022-04-10'),
(405, 'BrandE', 1094.36, '2019-05-20'),
(406, 'BrandF', 803.24, '2023-06-30'),
(407, 'BrandG', 1389.99, '2020-07-15'),
(408, 'BrandH', 1487.65, '2021-08-25'),
(409, 'BrandI', 1212.43, '2024-09-05'),
(410, 'BrandJ', 998.67, '2023-10-10');
```



```sql
INSERT INTO VirtualImage (VirtualImageID, Name, Age, Gender, ArtistID, SpecialistID, VoiceID) VALUES
(201, 'Kizuna', 200, 'Female', 101, 201, 301),
(202, 'Shiro', 28, 'Male', 102, 202, 302),
(203, 'Korone', 18, 'Female', 103, 203, 303),
(204, 'Hololive', 20, 'Female', 104, 204, 304),
(205, 'Hikaru', 15, 'Male', 105, 205, 305),
(206, 'Aqua', 17, 'Female', 106, 206, 306),
(207, 'Ren', 21, 'Male', 107, 207, 307),
(208, 'Fubuki', 19, 'Female', 108, 208, 308),
(209, 'Marine', 22, 'Female', 109, 209, 309),
(210, 'Yuki', 23, 'Male', 110, 210, 310);
```

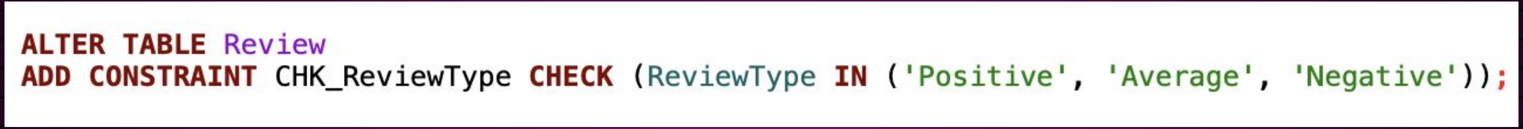


```sql
INSERT INTO RealPerson (StreamerID, FirstName, LastName, Age, Gender) VALUES
(101, 'Alice', 'Smith', 25, 'Female'),
(102, 'Bob', 'Johnson', 30, 'Male'),
(103, 'Charlie', 'Brown', 28, 'Male'),
(104, 'Diana', 'Evans', 26, 'Female'),
(105, 'Eve', 'Taylor', 24, 'Female'),
(106, 'Frank', 'Wright', 35, 'Male'),
(107, 'Grace', 'Hall', 29, 'Female'),
(108, 'Henry', 'Lee', 32, 'Male'),
(109, 'Ivy', 'Clark', 27, 'Female'),
(110, 'Jack', 'Walker', 33, 'Male');
```

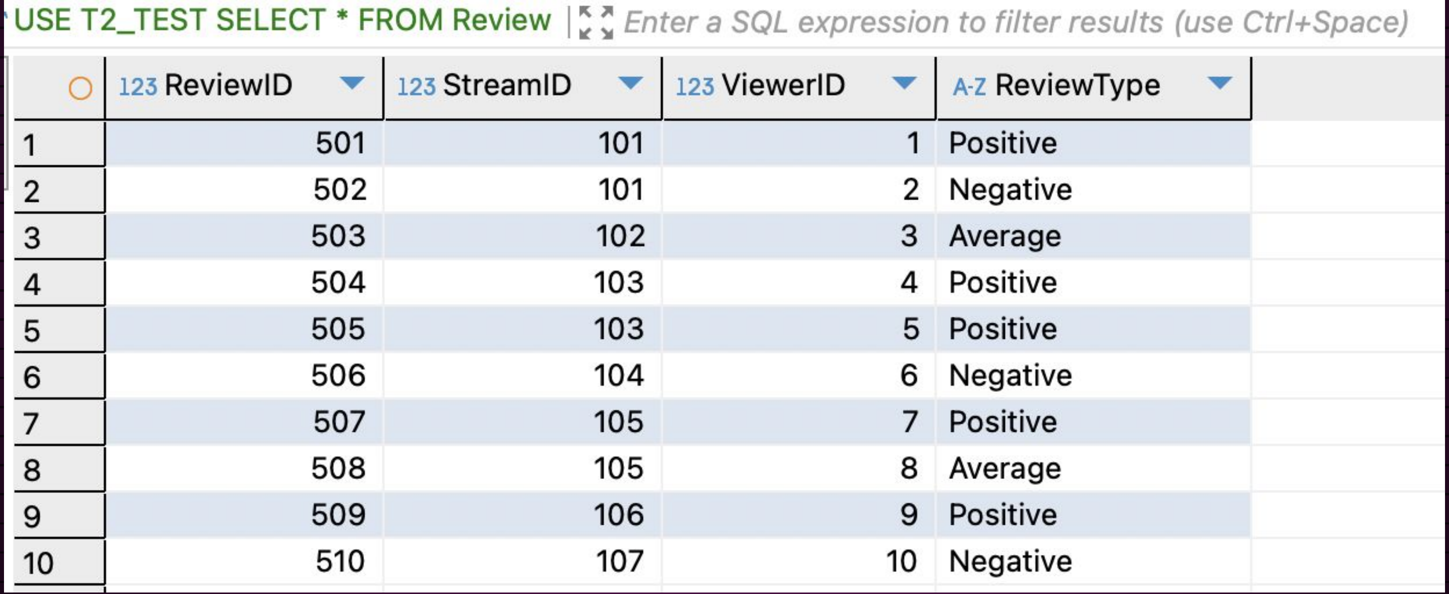


```sql
INSERT INTO VirtualIdol (VirtualIdolID, VirtualImageID, StreamerID) VALUES
(1, 201, 101),
(2, 202, 102),
(3, 203, 103),
(4, 204, 104),
(5, 205, 105),
(6, 206, 106),
(7, 207, 107),
(8, 208, 108),
(9, 209, 109),
(10, 210, 110);
```

Check Constraint



```
ALTER TABLE Review
ADD CONSTRAINT CHK_ReviewType CHECK (ReviewType IN ('Positive', 'Average', 'Negative'));
```



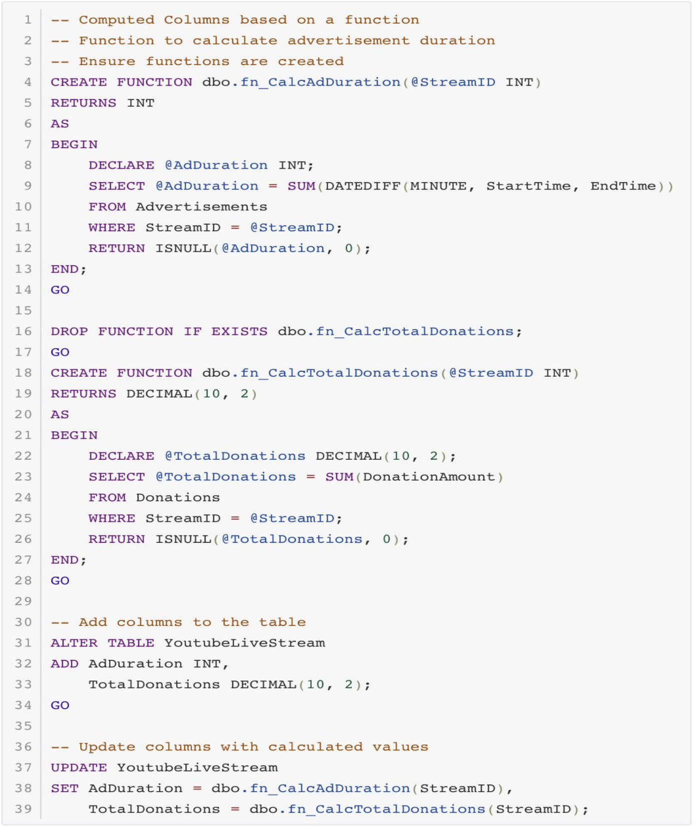
USE T2_TEST SELECT * FROM Review

○	123 ReviewID	123 StreamID	123 ViewerID	A-Z ReviewType
1	501	101	1	Positive
2	502	101	2	Negative
3	503	102	3	Average
4	504	103	4	Positive
5	505	103	5	Positive
6	506	104	6	Negative
7	507	105	7	Positive
8	508	105	8	Average
9	509	106	9	Positive
10	510	107	10	Negative

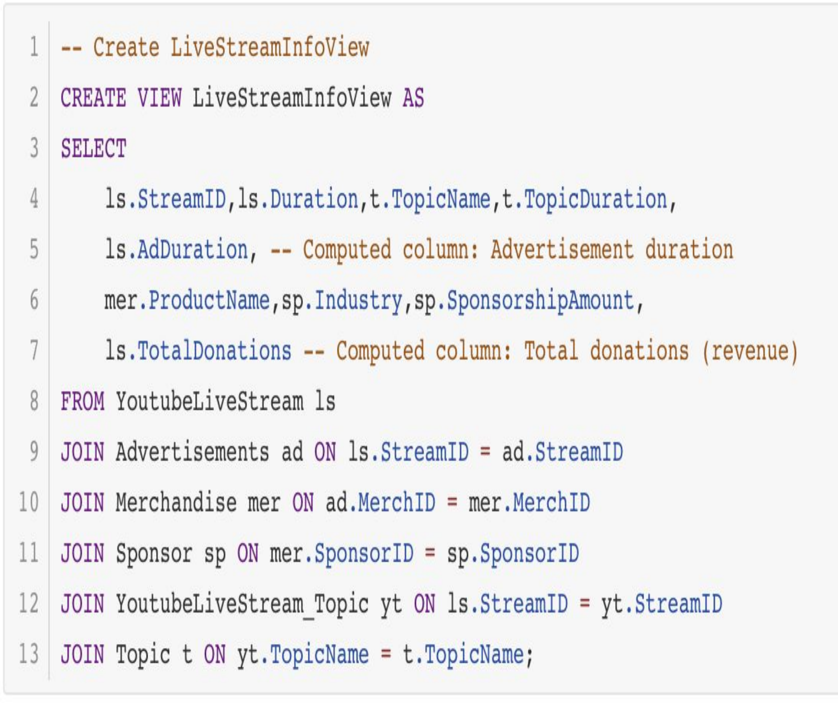
05

VIEW REPORT & ENCRYPTION

Live Stream Info View

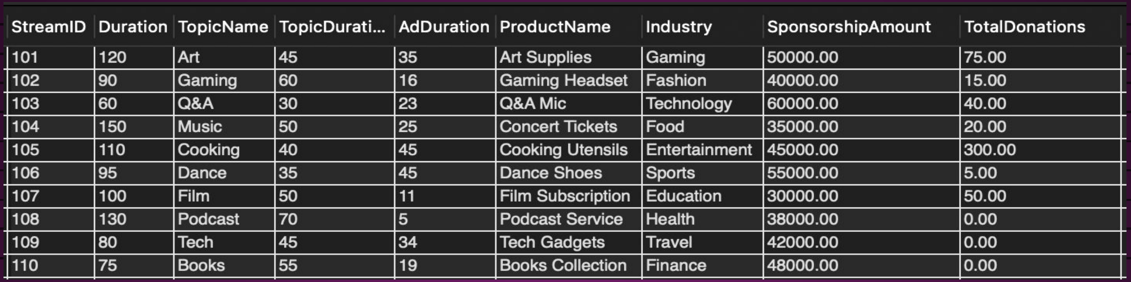


```sql
-- Computed Columns based on a function
-- Function to calculate advertisement duration
-- Ensure functions are created
CREATE FUNCTION dbo.fn_CalcAdDuration(@StreamID INT)
RETURNS INT
AS
BEGIN
    DECLARE @AdDuration INT;
    SELECT @AdDuration = SUM(DATEDIFF(MINUTE, StartTime, EndTime))
    FROM Advertisements
    WHERE StreamID = @StreamID;
    RETURN ISNULL(@AdDuration, 0);
END;
GO

DROP FUNCTION IF EXISTS dbo.fn_CalcTotalDonations;
GO
CREATE FUNCTION dbo.fn_CalcTotalDonations(@StreamID INT)
RETURNS DECIMAL(10, 2)
AS
BEGIN
    DECLARE @TotalDonations DECIMAL(10, 2);
    SELECT @TotalDonations = SUM(DonationAmount)
    FROM Donations
    WHERE StreamID = @StreamID;
    RETURN ISNULL(@TotalDonations, 0);
END;
GO

-- Add columns to the table
ALTER TABLE YoutubeLiveStream
ADD AdDuration INT,
    TotalDonations DECIMAL(10, 2);
GO

-- Update columns with calculated values
UPDATE YoutubeLiveStream
SET AdDuration = dbo.fn_CalcAdDuration(StreamID),
    TotalDonations = dbo.fn_CalcTotalDonations(StreamID);
```

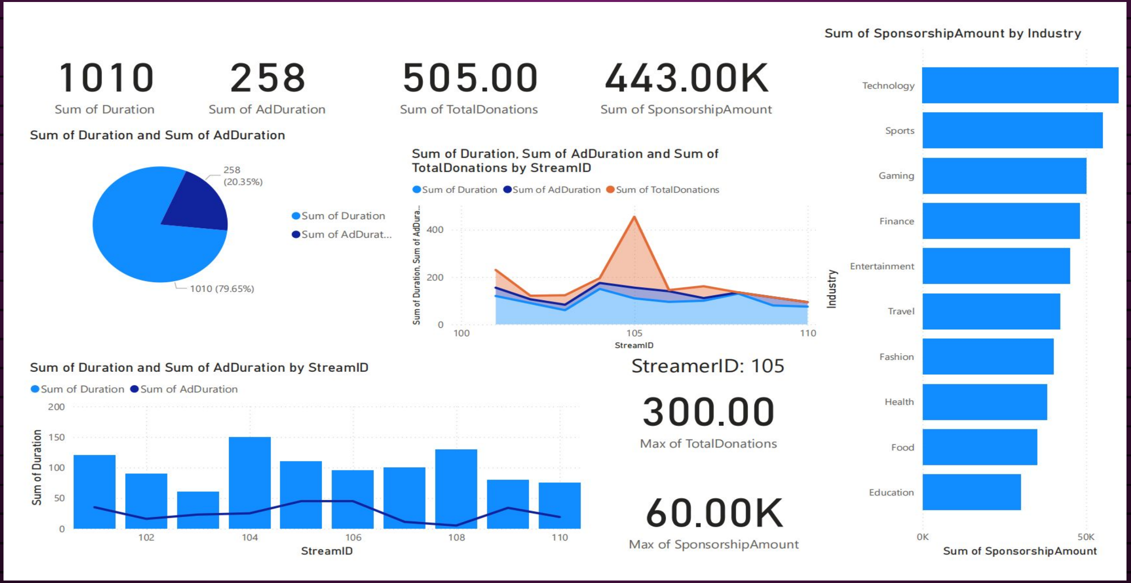


```sql
-- Create LiveStreamInfoView
CREATE VIEW LiveStreamInfoView AS
SELECT
    ls.StreamID,ls.Duration,t.TopicName,t.TopicDuration,
    ls.AdDuration, -- Computed column: Advertisement duration
    mer.ProductName,sp.Industry,sp.SponsorshipAmount,
    ls.TotalDonations -- Computed column: Total donations (revenue)
FROM YoutubeLiveStream ls
JOIN Advertisements ad ON ls.StreamID = ad.StreamID
JOIN Merchandise mer ON ad.MerchID = mer.MerchID
JOIN Sponsor sp ON mer.SponsorID = sp.SponsorID
JOIN YoutubeLiveStream_Topic yt ON ls.StreamID = yt.StreamID
JOIN Topic t ON yt.TopicName = t.TopicName;
```

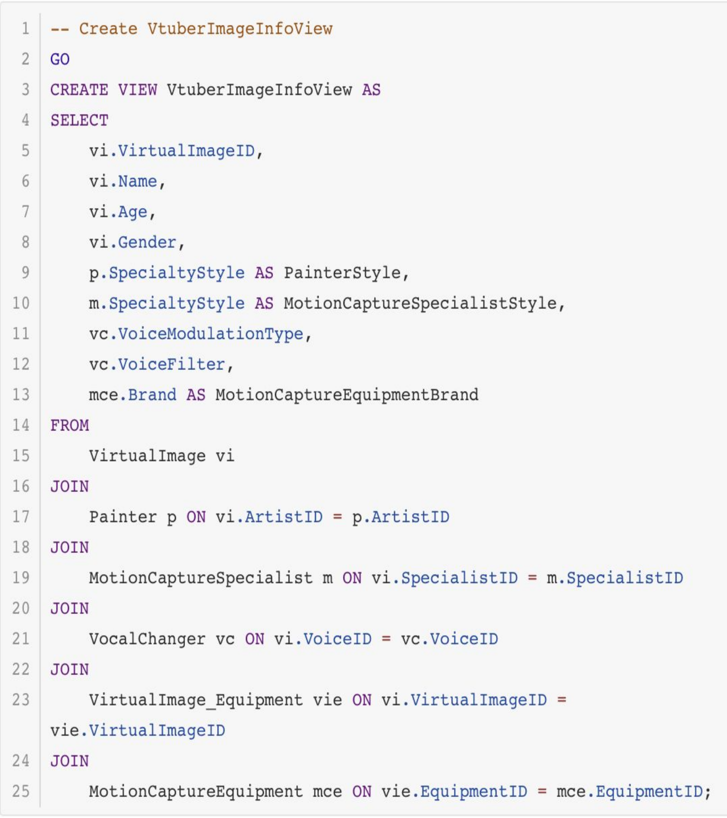
Live Stream Info View



StreamID	Duration	TopicName	TopicDurati...	AdDuration	ProductName	Industry	SponsorshipAmount	TotalDonations
101	120	Art	45	35	Art Supplies	Gaming	50000.00	75.00
102	90	Gaming	60	16	Gaming Headset	Fashion	40000.00	15.00
103	60	Q&A	30	23	Q&A Mic	Technology	60000.00	40.00
104	150	Music	50	25	Concert Tickets	Food	35000.00	20.00
105	110	Cooking	40	45	Cooking Utensils	Entertainment	45000.00	300.00
106	95	Dance	35	45	Dance Shoes	Sports	55000.00	5.00
107	100	Film	50	11	Film Subscription	Education	30000.00	50.00
108	130	Podcast	70	5	Podcast Service	Health	38000.00	0.00
109	80	Tech	45	34	Tech Gadgets	Travel	42000.00	0.00
110	75	Books	55	19	Books Collection	Finance	48000.00	0.00

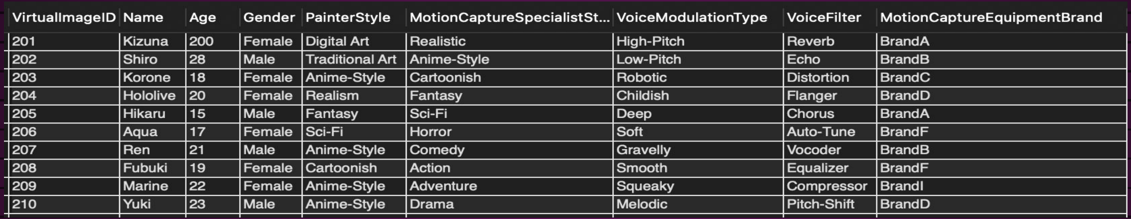


Vtuber Image Info View

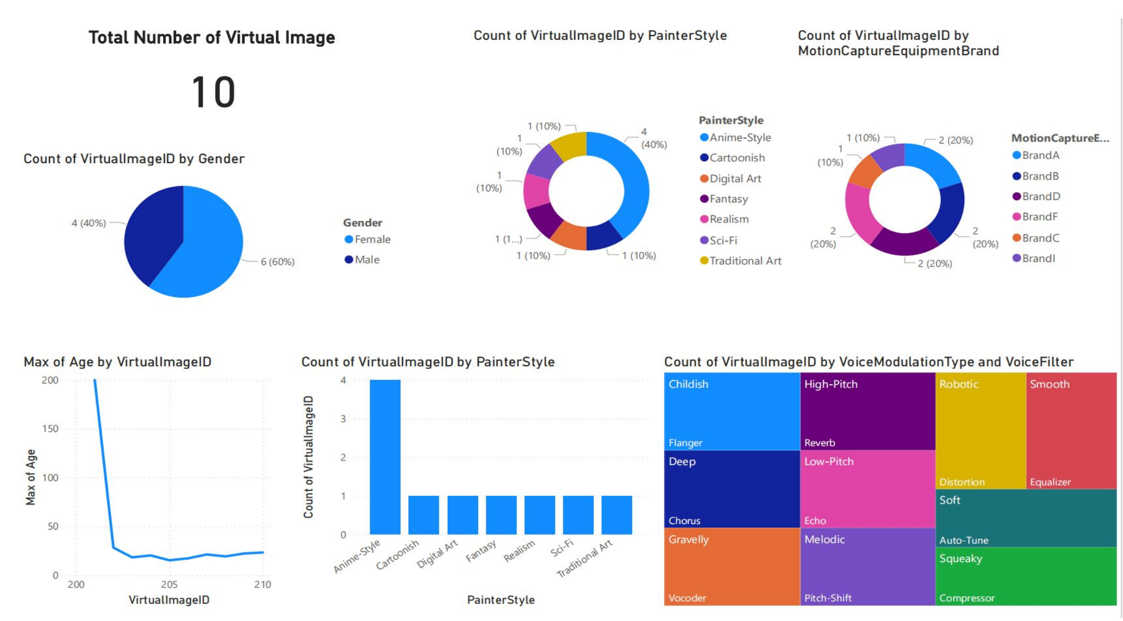


```sql
-- Create VtuberImageInfoView
GO
CREATE VIEW VtuberImageInfoView AS
SELECT
    vi.VirtualImageID,
    vi.Name,
    vi.Age,
    vi.Gender,
    p.SpecialtyStyle AS PainterStyle,
    m.SpecialtyStyle AS MotionCaptureSpecialistStyle,
    vc.VoiceModulationType,
    vc.VoiceFilter,
    mce.Brand AS MotionCaptureEquipmentBrand
FROM
    VirtualImage vi
JOIN
    Painter p ON vi.ArtistID = p.ArtistID
JOIN
    MotionCaptureSpecialist m ON vi.SpecialistID = m.SpecialistID
JOIN
    VocalChanger vc ON vi.VoiceID = vc.VoiceID
JOIN
    VirtualImage_Equipment vie ON vi.VirtualImageID =
vie.VirtualImageID
JOIN
    MotionCaptureEquipment mce ON vie.EquipmentID = mce.EquipmentID;
```

Vtuber Image Info View



VirtualImageID	Name	Age	Gender	PainterStyle	MotionCaptureSpecialistSt...	VoiceModulationType	VoiceFilter	MotionCaptureEquipmentBrand
201	Kizuna	200	Female	Digital Art	Realistic	High-Pitch	Reverb	BrandA
202	Shiro	28	Male	Traditional Art	Anime-Style	Low-Pitch	Echo	BrandB
203	Korone	18	Female	Anime-Style	Cartoonish	Robotic	Distortion	BrandC
204	Hololive	20	Female	Realism	Fantasy	Childish	Flanger	BrandD
205	Hikaru	15	Male	Fantasy	Sci-Fi	Deep	Chorus	BrandA
206	Aqua	17	Female	Sci-Fi	Horror	Soft	Auto-Tune	BrandF
207	Ren	21	Male	Anime-Style	Comedy	Gravelly	Vocoder	BrandB
208	Fubuki	19	Female	Cartoonish	Action	Smooth	Equalizer	BrandF
209	Marine	22	Female	Anime-Style	Adventure	Squeaky	Compressor	BrandI
210	Yuki	23	Male	Anime-Style	Drama	Melodic	Pitch-Shift	BrandD



Total Number of Virtual Image

10

Count of VirtualImageID by PainterStyle

Count of VirtualImageID by MotionCaptureEquipmentBrand

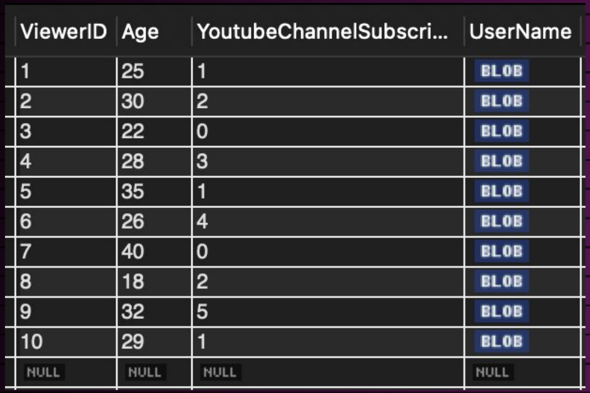
Count of VirtualImageID by Gender

Max of Age by VirtualImageID

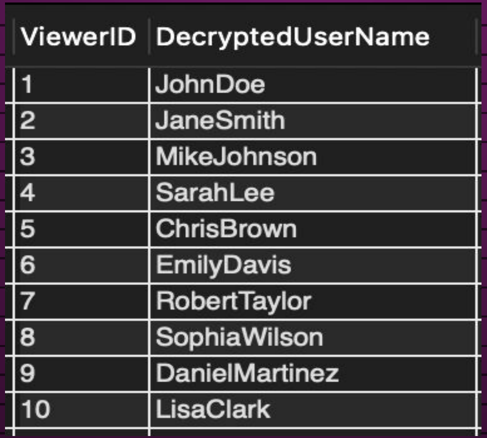
Count of VirtualImageID by PainterStyle

Count of VirtualImageID by VoiceModulationType and VoiceFilter

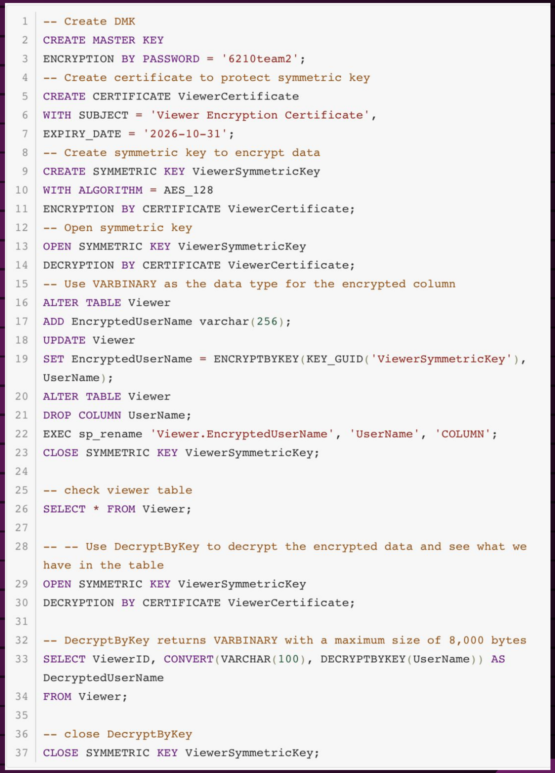
Database Encryption



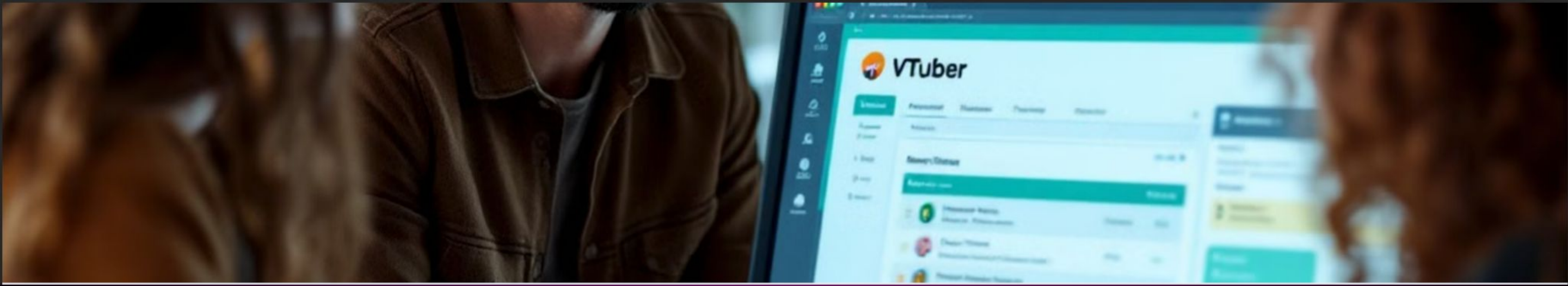
ViewerID	Age	YoutubeChannelSubscri...	UserName
1	25	1	BLOB
2	30	2	BLOB
3	22	0	BLOB
4	28	3	BLOB
5	35	1	BLOB
6	26	4	BLOB
7	40	0	BLOB
8	18	2	BLOB
9	32	5	BLOB
10	29	1	BLOB
NULL	NULL	NULL	NULL



ViewerID	DecryptedUserName
1	JohnDoe
2	JaneSmith
3	MikeJohnson
4	SarahLee
5	ChrisBrown
6	EmilyDavis
7	RobertTaylor
8	SophiaWilson
9	DanielMartinez
10	LisaClark

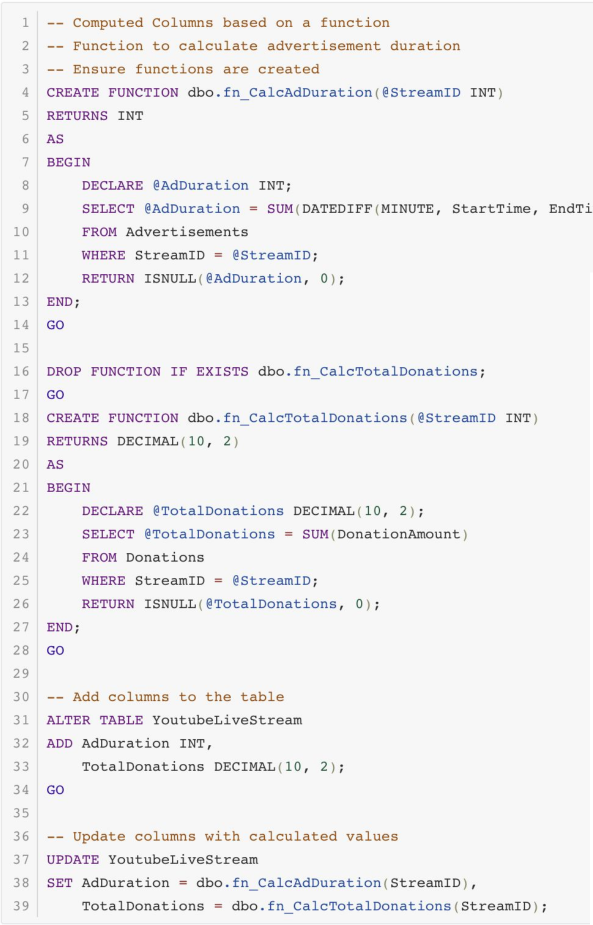


```sql
-- Create DMK
CREATE MASTER KEY
ENCRYPTION BY PASSWORD = '6210team2';
-- Create certificate to protect symmetric key
CREATE CERTIFICATE ViewerCertificate
WITH SUBJECT = 'Viewer Encryption Certificate',
EXPIRY_DATE = '2026-10-31';
-- Create symmetric key to encrypt data
CREATE SYMMETRIC KEY ViewerSymmetricKey
WITH ALGORITHM = AES_128
ENCRYPTION BY CERTIFICATE ViewerCertificate;
-- Open symmetric key
OPEN SYMMETRIC KEY ViewerSymmetricKey
DECRYPTION BY CERTIFICATE ViewerCertificate;
-- Use VARBINARY as the data type for the encrypted column
ALTER TABLE Viewer
ADD EncryptedUserName varchar(256);
UPDATE Viewer
SET EncryptedUserName = ENCRYPTBYKEY(KEY_GUID('ViewerSymmetricKey'), UserName);
ALTER TABLE Viewer
DROP COLUMN UserName;
EXEC sp_rename 'Viewer.EncryptedUserName', 'UserName', 'COLUMN';
CLOSE SYMMETRIC KEY ViewerSymmetricKey;

-- check viewer table
SELECT * FROM Viewer;

-- -- Use DecryptByKey to decrypt the encrypted data and see what we have in the table
OPEN SYMMETRIC KEY ViewerSymmetricKey
DECRYPTION BY CERTIFICATE ViewerCertificate;

-- DecryptByKey returns VARBINARY with a maximum size of 8,000 bytes
SELECT ViewerID, CONVERT(VARCHAR(100), DECRYPTBYKEY(UserName)) AS DecryptedUserName
FROM Viewer;

-- close DecryptByKey
CLOSE SYMMETRIC KEY ViewerSymmetricKey;
```



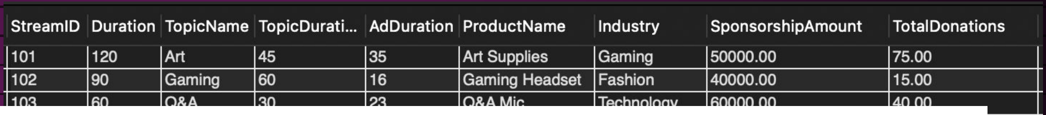
Q & A

View Report

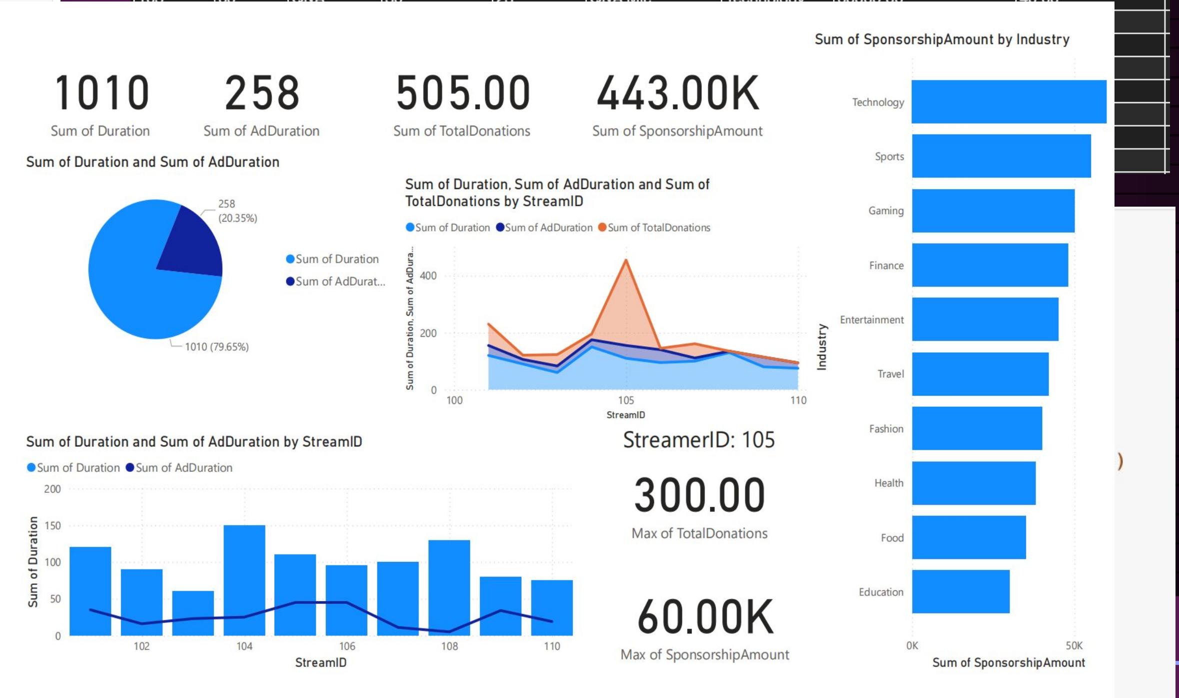


```sql
-- Computed Columns based on a function
-- Function to calculate advertisement duration
-- Ensure functions are created
CREATE FUNCTION dbo.fn_CalcAdDuration(@StreamID INT)
RETURNS INT
AS
BEGIN
    DECLARE @AdDuration INT;
    SELECT @AdDuration = SUM(DATEDIFF(MINUTE, StartTime, EndTime))
    FROM Advertisements
    WHERE StreamID = @StreamID;
    RETURN ISNULL(@AdDuration, 0);
END;
GO

DROP FUNCTION IF EXISTS dbo.fn_CalcTotalDonations;
GO
CREATE FUNCTION dbo.fn_CalcTotalDonations(@StreamID INT)
RETURNS DECIMAL(10, 2)
AS
BEGIN
    DECLARE @TotalDonations DECIMAL(10, 2);
    SELECT @TotalDonations = SUM(DonationAmount)
    FROM Donations
    WHERE StreamID = @StreamID;
    RETURN ISNULL(@TotalDonations, 0);
END;
GO

-- Add columns to the table
ALTER TABLE YoutubeLiveStream
ADD AdDuration INT,
    TotalDonations DECIMAL(10, 2);
GO

-- Update columns with calculated values
UPDATE YoutubeLiveStream
SET AdDuration = dbo.fn_CalcAdDuration(StreamID),
    TotalDonations = dbo.fn_CalcTotalDonations(StreamID);
```

Live Stream Info View



StreamID	Duration	TopicName	TopicDurati...	AdDuration	ProductName	Industry	SponsorshipAmount	TotalDonations
101	120	Art	45	35	Art Supplies	Gaming	50000.00	75.00
102	90	Gaming	60	16	Gaming Headset	Fashion	40000.00	15.00
103	60	Q&A	30	23	Q&A Mic	Technology	60000.00	40.00



1010
Sum of Duration

258
Sum of AdDuration

505.00
Sum of TotalDonations

443.00K
Sum of SponsorshipAmount

Sum of Duration and Sum of AdDuration

Sum of Duration, Sum of AdDuration and Sum of TotalDonations by StreamID

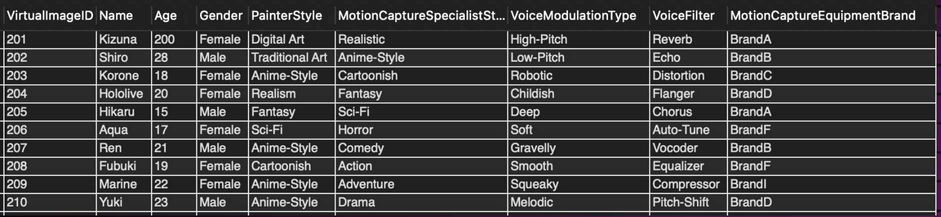
Sum of SponsorshipAmount by Industry

Sum of Duration and Sum of AdDuration by StreamID

StreamerID: 105

300.00
Max of TotalDonations

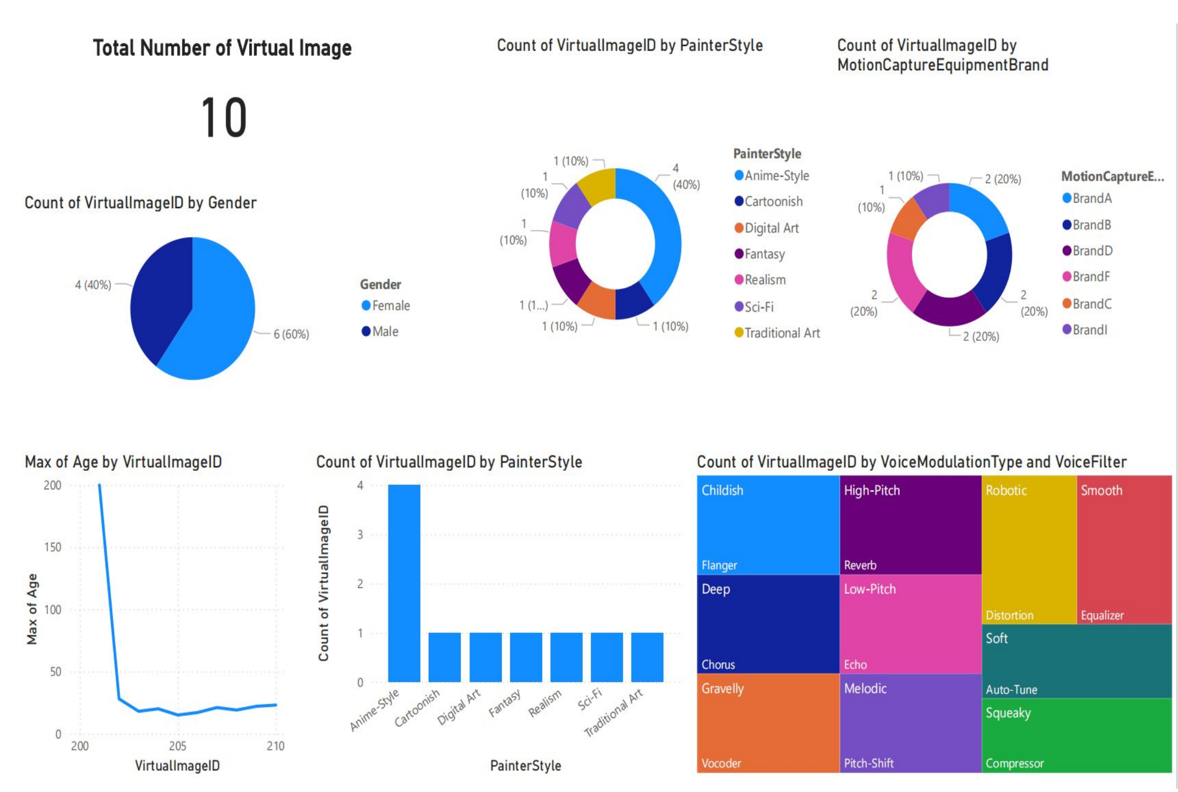
60.00K
Max of SponsorshipAmount



VirtualImageID	Name	Age	Gender	PainterStyle	MotionCaptureSpecialistSt...	VoiceModulationType	VoiceFilter	MotionCaptureEquipmentBrand
201	Kizuna	200	Female	Digital Art	Realistic	High-Pitch	Reverb	BrandA
202	Shiro	28	Male	Traditional Art	Anime-Style	Low-Pitch	Echo	BrandB
203	Korone	18	Female	Anime-Style	Cartoonish	Robotic	Distortion	BrandC
204	Hololive	20	Female	Realism	Fantasy	Childish	Flanger	BrandD
205	Hikaru	15	Male	Fantasy	Sci-Fi	Deep	Chorus	BrandA
206	Aqua	17	Female	Sci-Fi	Horror	Soft	Auto-Tune	BrandF
207	Ren	21	Male	Anime-Style	Comedy	Gravelly	Vocoder	BrandB
208	Fubuki	19	Female	Cartoonish	Action	Smooth	Equalizer	BrandF
209	Marine	22	Female	Anime-Style	Adventure	Squeaky	Compressor	BrandI
210	Yuki	23	Male	Anime-Style	Drama	Melodic	Pitch-Shift	BrandD

View Report

Vtuber Image Info View



Total Number of Virtual Image

10

Count of VirtualImageID by PainterStyle

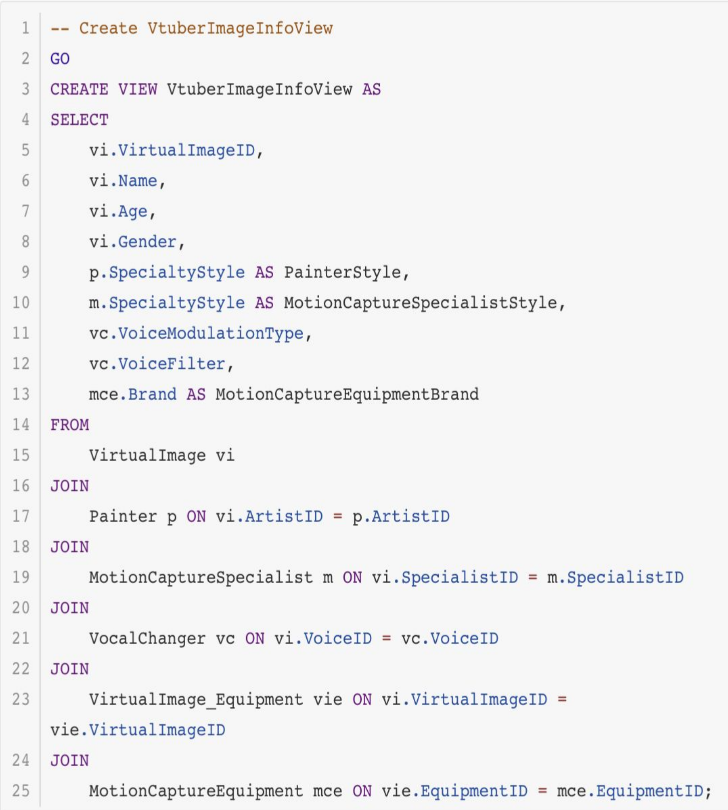
Count of VirtualImageID by MotionCaptureEquipmentBrand

Count of VirtualImageID by Gender

Max of Age by VirtualImageID

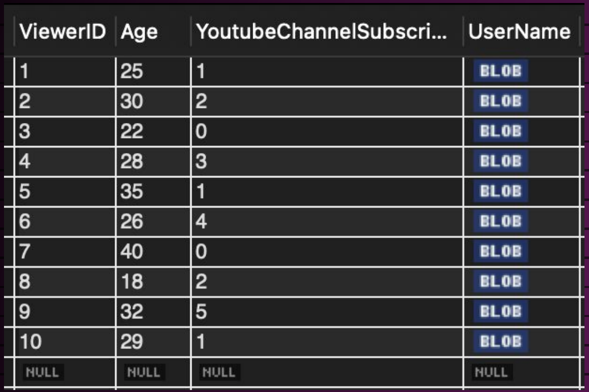
Count of VirtualImageID by PainterStyle

Count of VirtualImageID by VoiceModulationType and VoiceFilter

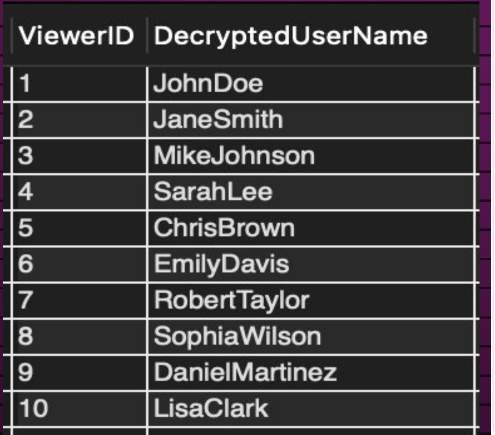


```sql
-- Create VtuberImageInfoView
GO
CREATE VIEW VtuberImageInfoView AS
SELECT
    vi.VirtualImageID,
    vi.Name,
    vi.Age,
    vi.Gender,
    p.SpecialtyStyle AS PainterStyle,
    m.SpecialtyStyle AS MotionCaptureSpecialistStyle,
    vc.VoiceModulationType,
    vc.VoiceFilter,
    mce.Brand AS MotionCaptureEquipmentBrand
FROM
    VirtualImage vi
JOIN
    Painter p ON vi.ArtistID = p.ArtistID
JOIN
    MotionCaptureSpecialist m ON vi.SpecialistID = m.SpecialistID
JOIN
    VocalChanger vc ON vi.VoiceID = vc.VoiceID
JOIN
    VirtualImage_Equipment vie ON vi.VirtualImageID =
vie.VirtualImageID
JOIN
    MotionCaptureEquipment mce ON vie.EquipmentID = mce.EquipmentID;
```

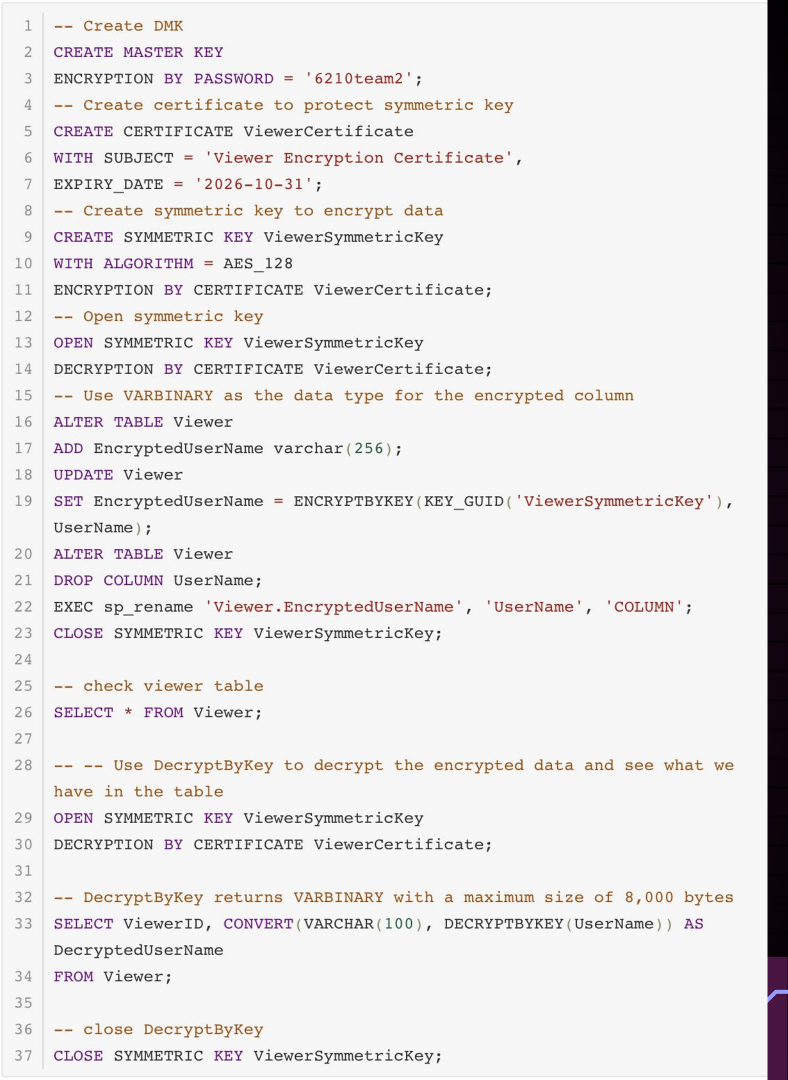
Database Encryption



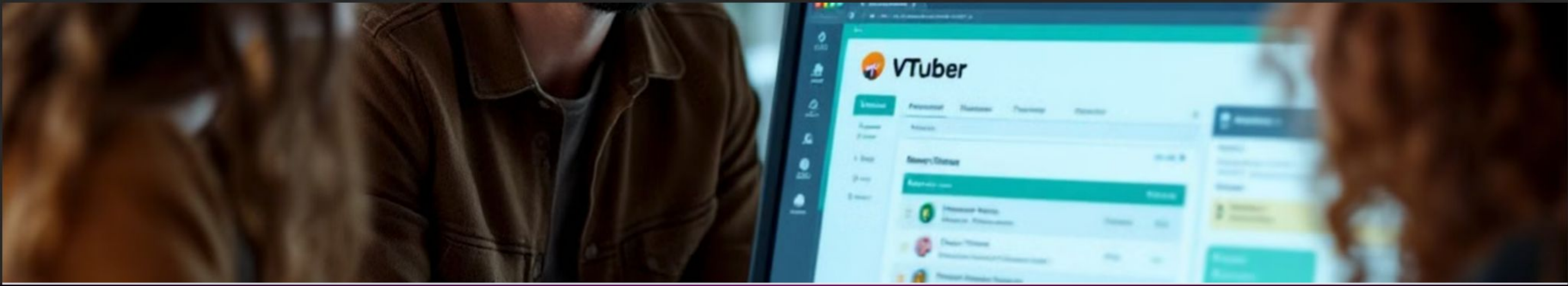
ViewerID	Age	YoutubeChannelSubscri...	UserName
1	25	1	BLOB
2	30	2	BLOB
3	22	0	BLOB
4	28	3	BLOB
5	35	1	BLOB
6	26	4	BLOB
7	40	0	BLOB
8	18	2	BLOB
9	32	5	BLOB
10	29	1	BLOB
NULL	NULL	NULL	NULL



ViewerID	DecryptedUserName
1	JohnDoe
2	JaneSmith
3	MikeJohnson
4	SarahLee
5	ChrisBrown
6	EmilyDavis
7	RobertTaylor
8	SophiaWilson
9	DanielMartinez
10	LisaClark



```sql
-- Create DMK
CREATE MASTER KEY
ENCRYPTION BY PASSWORD = '6210team2';
-- Create certificate to protect symmetric key
CREATE CERTIFICATE ViewerCertificate
WITH SUBJECT = 'Viewer Encryption Certificate',
EXPIRY_DATE = '2026-10-31';
-- Create symmetric key to encrypt data
CREATE SYMMETRIC KEY ViewerSymmetricKey
WITH ALGORITHM = AES_128
ENCRYPTION BY CERTIFICATE ViewerCertificate;
-- Open symmetric key
OPEN SYMMETRIC KEY ViewerSymmetricKey
DECRYPTION BY CERTIFICATE ViewerCertificate;
-- Use VARBINARY as the data type for the encrypted column
ALTER TABLE Viewer
ADD EncryptedUserName varchar(256);
UPDATE Viewer
SET EncryptedUserName = ENCRYPTBYKEY(KEY_GUID('ViewerSymmetricKey'), UserName);
ALTER TABLE Viewer
DROP COLUMN UserName;
EXEC sp_rename 'Viewer.EncryptedUserName', 'UserName', 'COLUMN';
CLOSE SYMMETRIC KEY ViewerSymmetricKey;

-- check viewer table
SELECT * FROM Viewer;

-- -- Use DecryptByKey to decrypt the encrypted data and see what we have in the table
OPEN SYMMETRIC KEY ViewerSymmetricKey
DECRYPTION BY CERTIFICATE ViewerCertificate;

-- DecryptByKey returns VARBINARY with a maximum size of 8,000 bytes
SELECT ViewerID, CONVERT(VARCHAR(100), DECRYPTBYKEY(UserName)) AS DecryptedUserName
FROM Viewer;

-- close DecryptByKey
CLOSE SYMMETRIC KEY ViewerSymmetricKey;
```



Q & A