BlogPlatformDB

Kurs : SQL
Name : Mahboubeh Ranjbar
Dozent: Herr Lev A. Brodski
Datum : 14.08.2024

Projektziel:

Das BlogPlatformDB-Projekt hat das Ziel, eine robuste und skalierbare Datenbankstruktur für eine Blogging-Plattform zu entwickeln. Diese Plattform ermöglicht die Verwaltung von Blog-Posts, Benutzern, Kommentaren, Kategorien, Tags und zahlungsbezogenen Informationen. Zudem implementiert sie wichtige Geschäftslogik und sorgt für die Datenintegrität durch den Einsatz von gespeicherten Prozeduren, Funktionen und Triggern.

Hauptfunktionen:

- Blog-Management: Verwaltung von Blog-Posts mit zugehörigen Autoren, Kategorien, und Tags.
- Benutzer- und Rollenverwaltung: Verwaltung von Benutzerkonten, Rollen und VIP-Status, sowie Abonnementinformationen.
- Interaktionsverwaltung: Erfassung und Verwaltung von Kommentaren, Likes, und gespeicherten Posts.
- Zahlungsverwaltung: Verwaltung und Nachverfolgung von Benutzerzahlungen und VIP-Status.

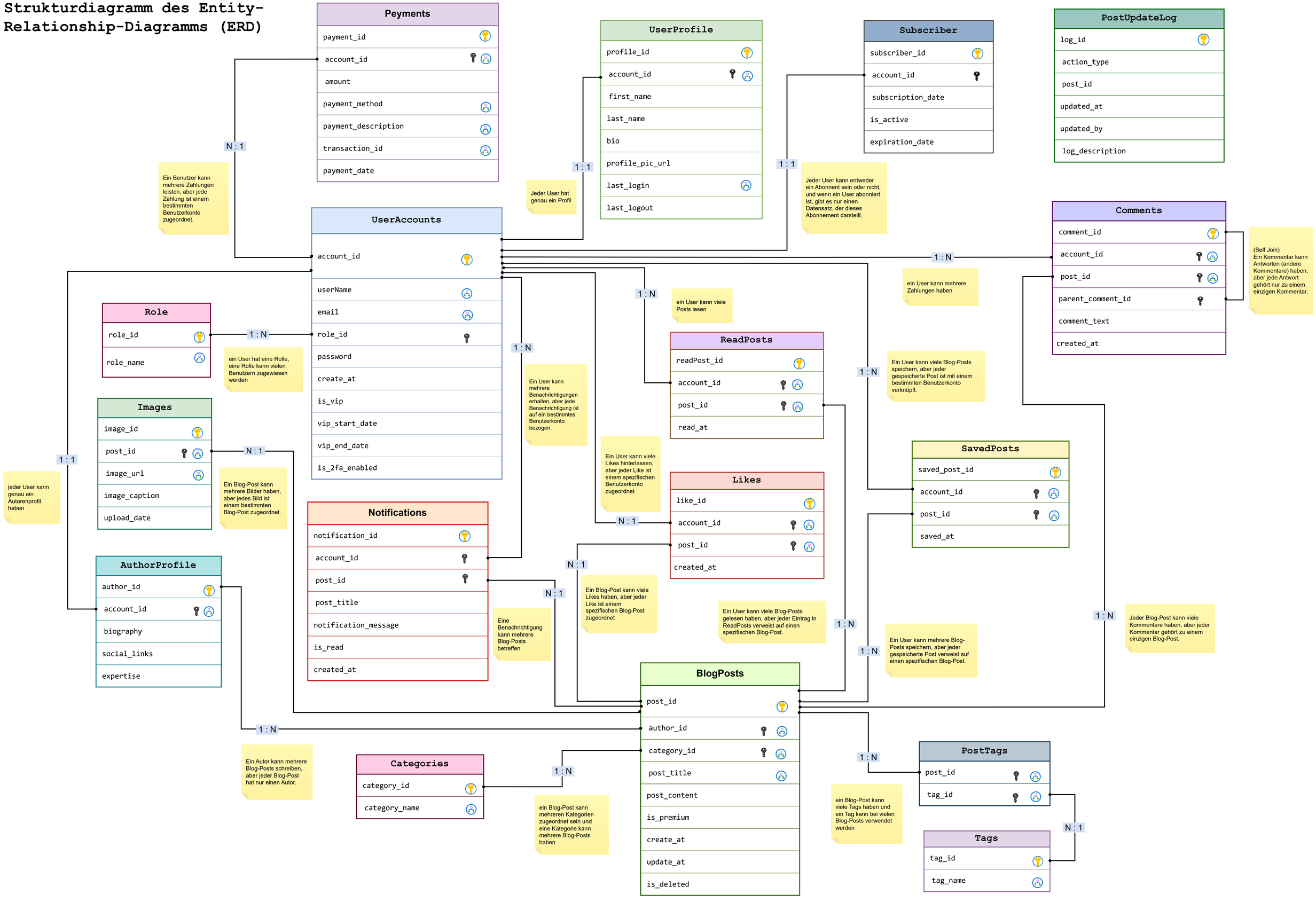
Technologien:

- SQL Server Management Studio (SSMS): Hauptwerkzeug zur Entwicklung, Verwaltung und Analyse der Datenbank.
- T-SQL: Programmiersprache für die Entwicklung der gespeicherten Prozeduren, Funktionen und Trigger.

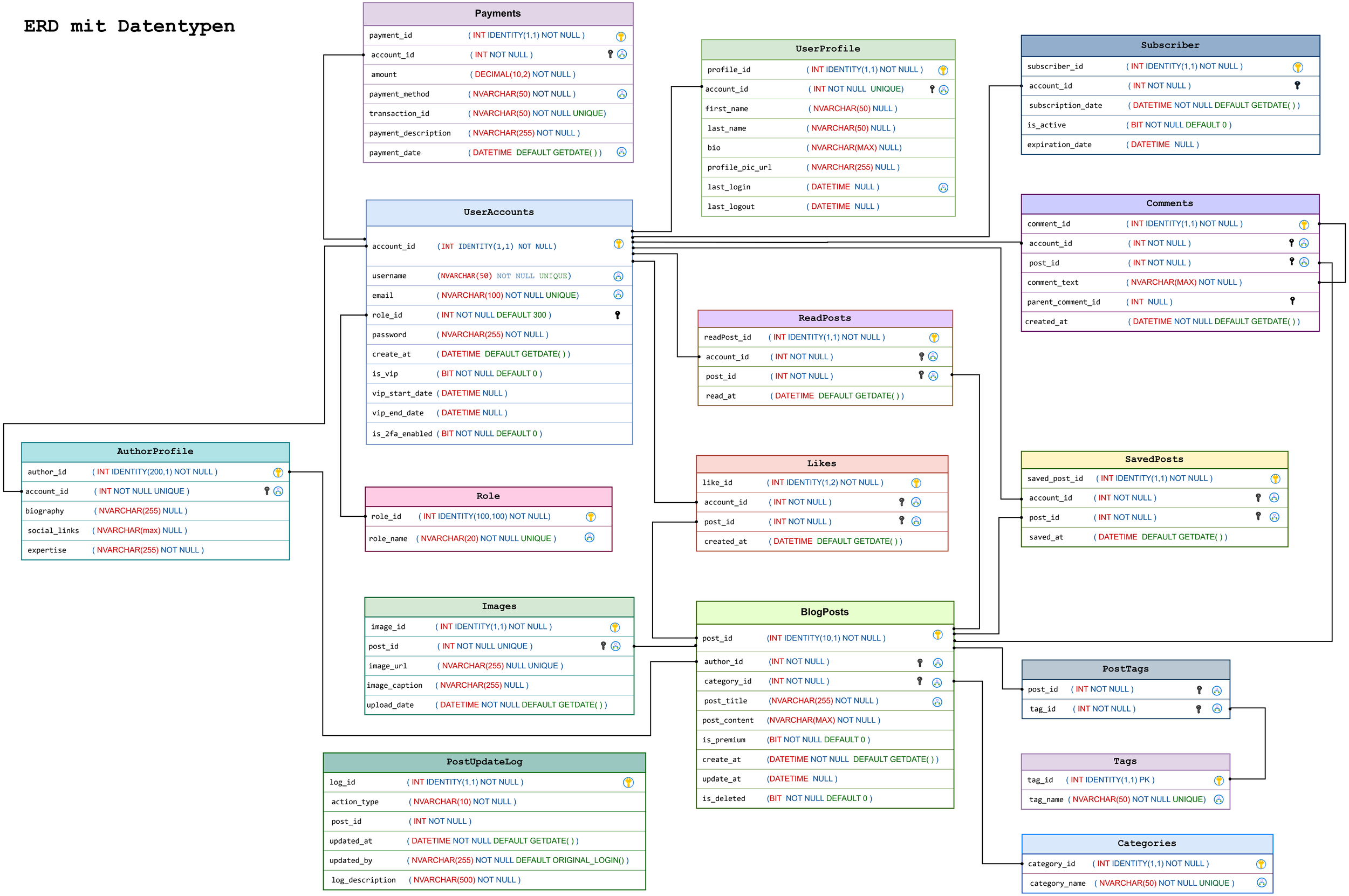
Datenbankkomponenten:

- **Tabellen**: Strukturierte Speicherung der Daten, einschließlich Beziehungen zwischen Blog-Posts, Benutzern, Kommentaren, Kategorien und Zahlungen.
- **Gespeicherte Prozeduren**: Implementieren von Geschäftslogik zur Verwaltung von Kommentaren, Benachrichtigungen, Benutzer-Upgrades und Backups.
- **Funktionen**: Bereitstellung von wiederverwendbaren SQL-Ausdrücken zur Berechnung von Metriken wie Kommentaranzahl und Blog-Post-Beliebtheit.
- **Trigger**: Automatisches Auslösen von Aktionen bei Datenänderungen zur Sicherstellung der Datenintegrität und Protokollierung von Änderungen.

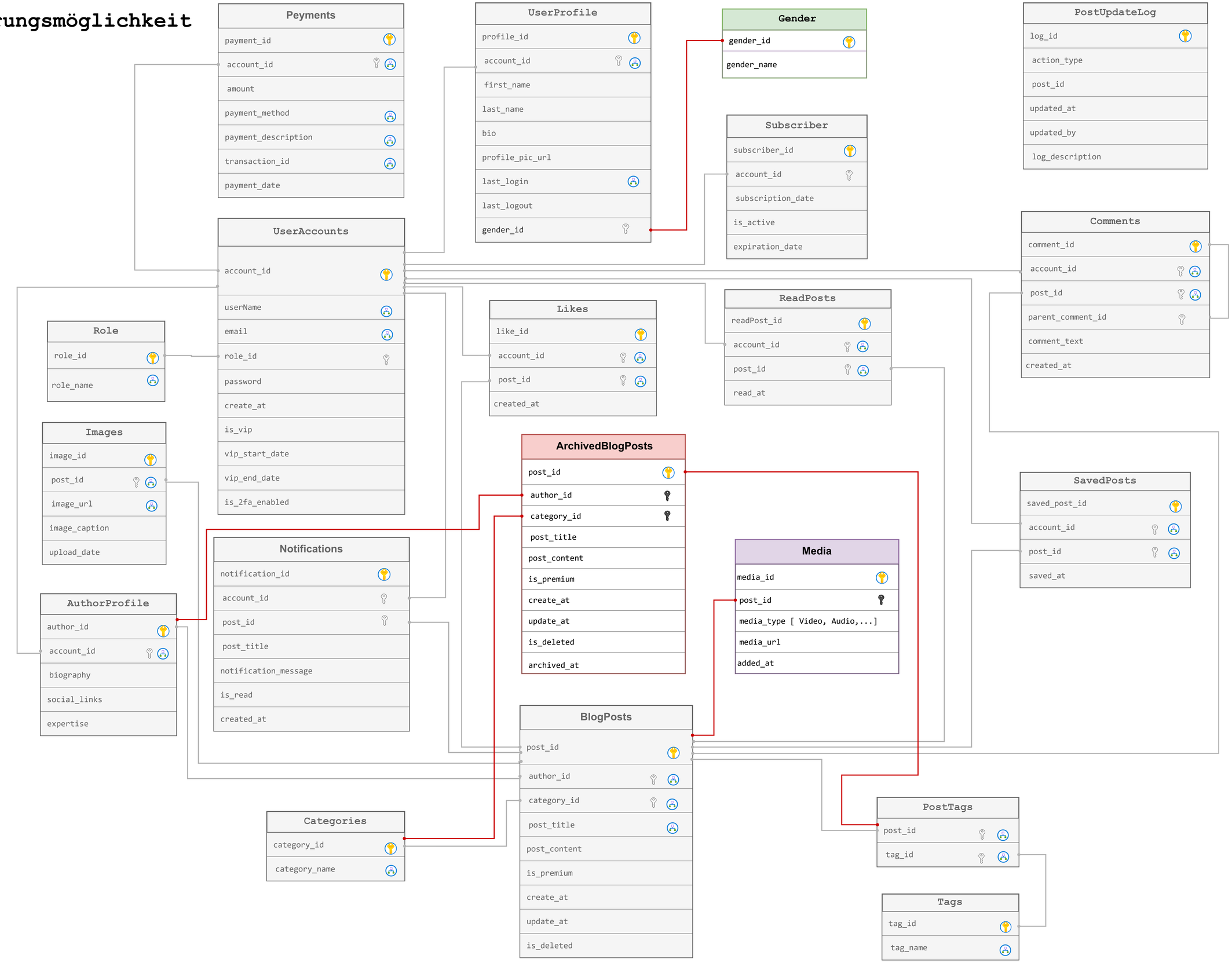
Strukturdiagramm des Entity-Relationship-Diagramms (ERD)



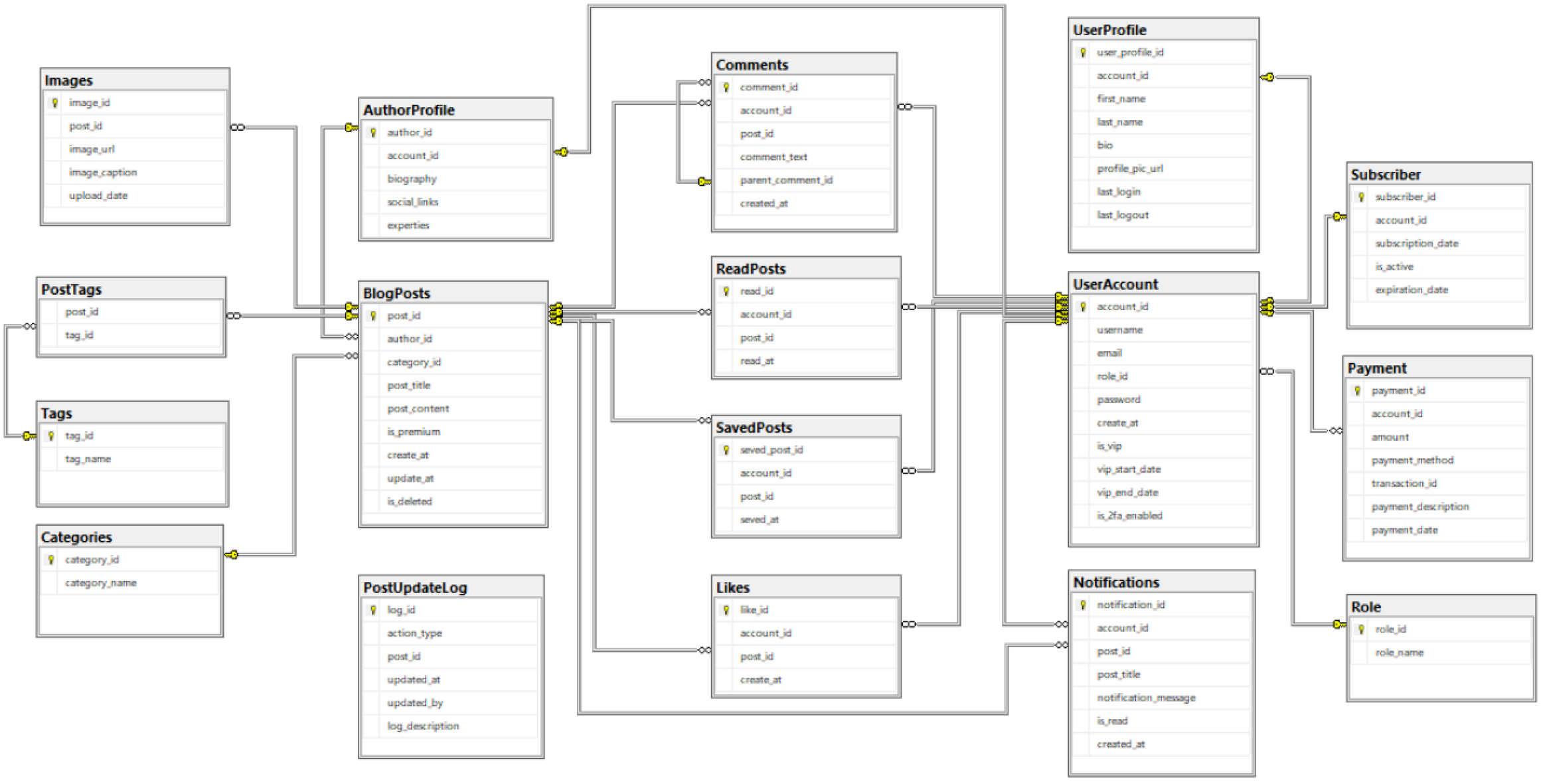
ERD mit Datentypen



Erweiterungsmöglichkeit



ERD aus SQL Server Management Studio (SSMS)



Tables

BlogPosts:

- **Zweck:** Speicherung von Blog-Inhalten, die von Autoren erstellt wurden. Enthält Felder wie Titel, Inhalt, Kategorie, Erstellungs- und Aktualisierungsdaten.

UserAccount:

- **Zweck:** Diese Tabelle speichert grundlegende Informationen über die Benutzer der Blog-Plattform, einschließlich Anmeldedaten und Status (z.B. ob der Benutzer ein VIP ist).

UserProfile:

- **Zweck:** Speichert zusätzliche Profildetails der Benutzer, wie z.B. ihren Namen, Biografie und Profilbild.

AuthorProfile:

- **Zweck:** Speichert detaillierte Informationen über Autoren auf der Plattform.

Comments:

- **Zweck:** Speicherung der Benutzerkommentare zu Blog-Posts.

Categories:

- **Zweck:** Verwaltung der verschiedenen Kategorien, denen Blog-Posts zugeordnet werden können.

Tags und PostTags:

- **Zweck:** Verwaltung von Tags, die Blog-Posts zugeordnet werden können, zur besseren Organisation und Suche.

Subscriber:

- **Zweck:** Verwaltung von Benutzern, die sich für Updates oder Newsletter anmelden.

Payment:

- **Zweck:** Speicherung von Zahlungsinformationen für VIP-Mitgliedschaften.

Notifications:

- **Zweck:** Speichert Benachrichtigungen, die den Benutzern der Plattform gesendet werden.

PostUpdateLog:

- **Zweck:** Protokolliert alle Änderungen, die an Blog-Beiträgen vorgenommen werden.

Images:

- **Zweck:** Speichert Bilder, die in Blog-Beiträgen verwendet werden.

Likes:

- **Zweck:** Speichert Informationen über die Likes, die Benutzer auf Blog-Beiträge vergeben haben.

SavedPosts:

- **Zweck:** Speichert Blog-Beiträge, die von Benutzern für das spätere Lesen gespeichert wurden.

ReadPosts:

- **Zweck:** Protokolliert, welche Blog-Beiträge von welchen Benutzern gelesen wurden.

Views

View_BlogPosts_With_Author_Likes_Comments:

- Zeigt Blog-Posts zusammen mit dem Autorennamen, der Anzahl der Likes und der Anzahl der Kommentare für jeden Post. Nützlich zur Bewertung der Beliebtheit und Interaktion eines Posts.
- **SQL-Konstrukte:** INNER JOIN, LEFT JOIN, GROUP BY, COUNT, DISTINCT

View_Authors_With_More_Than_Two_Posts:

- Zeigt Autoren, die mehr als zwei Blog-Posts veröffentlicht haben, um aktive Autoren hervorzuheben.
- **SQL-Konstrukte:** INNER JOIN, GROUP BY, COUNT, HAVING

View_Comments_With_Replies:

- Zeigt Kommentare zusammen mit ihren Antworten (verschachtelte Kommentare),um Diskussionen unter einem Blog-Post zu verfolgen.
- **SQL-Konstrukte:** LEFT JOIN, SELF JOIN

View_Posts_With_Comments_And_User_Details:

- Zeigt Blog-Posts zusammen mit den Details der Benutzer, die Kommentare hinterlassen haben. Hilfreich für die Analyse der Benutzerinteraktionen.
- **SQL-Konstrukte:** INNER JOIN

View_Subscriber_With_Email:

- Zeigt alle Abonnenten mit ihren E-Mail-Adressen. Nützlich für das Management von Benachrichtigungen und Newslettern.
- **SQL-Konstrukte:** INNER JOIN

View_Tags_With_PostCounts:

- Zeigt Tags mit der Anzahl der zugeordneten Posts, um häufig verwendete Tags und beliebte Themen zu analysieren.
- **SQL-Konstrukte:** INNER JOIN, GROUP BY, COUNT

View_UserAccount_With_Payment_And_VIP_Status:

- Zeigt Benutzerkonten mit Zahlungsinformationen und VIP-Status, um Benutzeraktivitäten und Zahlungsverhalten zu überwachen.
- **SQL-Konstrukte:** FULL JOIN

View_User_Comment_And_Like_Counts:

- Zeigt die Anzahl der Kommentare und Likes jedes Benutzers. Hilft bei der Analyse des Benutzerengagements.
- **SQL-Konstrukte:** INNER JOIN, GROUP BY, COUNT, DISTINCT

Scalar Functions(SF)

SF_Get_Like_Count_By_Post:

Gibt die Anzahl der Likes für einen bestimmten Blog-Post zurück. Diese Funktion ist nützlich, um die Beliebtheit eines Beitrags zu bewerten.

SF_Check_Account_Exists:

Überprüft, ob ein Benutzerkonto basierend auf einer bestimmten Account-ID existiert. Diese Funktion kann verwendet werden, um die Existenz eines Kontos vor der Durchführung weiterer Operationen zu verifizieren.

SF_Get_Post_Count_By_Category:

Gibt die Anzahl der Blog-Posts in einer bestimmten Kategorie zurück. Diese Funktion hilft, die Anzahl der Beiträge zu analysieren, die verschiedenen Themen oder Kategorien zugeordnet sind.

SF_Has_User_Read_Post:

Überprüft, ob ein Benutzer einen bestimmten Blog-Post gelesen hat. Diese Funktion ist nützlich, um zu steuern, ob ein Benutzer bestimmte Aktionen wie das Hinzufügen eines Kommentars ausführen darf.

Table-Valued Functions(TF)

TF_Get_Comment_And_Like_Count:

Gibt eine Tabelle mit der Anzahl der Kommentare und Likes für jeden Blog-Post zurück. Diese Funktion hilft bei der Analyse von Benutzerinteraktionen und der Beliebtheit von Beiträgen.

TF_Get_Most_Like_Post_By_Category:

Gibt den Blog-Post mit den meisten Likes in einer bestimmten Kategorie zurück. Diese Funktion ist nützlich, um die erfolgreichsten Beiträge innerhalb einer Kategorie zu identifizieren.

TF_Get_Posts_By_Author:

Gibt eine Tabelle mit allen Blog-Posts eines bestimmten Autors zurück. Diese Funktion unterstützt die Analyse der Aktivitäten und Beiträge einzelner Autoren.

TF_Get_Tags_By_Post:

Gibt alle Tags zurück, die einem bestimmten Blog-Post zugeordnet sind. Diese Funktion ist nützlich, um die Themen und Schlagworte eines Beitrags zu ermitteln.

Stored Procedures (SP)

SP_Add_Comment:

Diese Prozedur ermöglicht es einem Benutzer, einen Kommentar zu einem Blog-Post hinzuzufügen, nachdem überprüft wurde, ob der Benutzer den Beitrag gelesen hat. Zusätzlich wird sichergestellt, dass der Kommentartext nicht leer ist und die maximale Länge nicht überschreitet.

SP_Upgrade_User_To_VIP:

Diese Prozedur aktualisiert das Konto eines Benutzers auf den VIP-Status, nachdem eine erfolgreiche Zahlung erfolgt ist. Sie aktualisiert die VIP-Daten des Benutzers und erstellt einen entsprechenden Zahlungseintrag in der Datenbank.

SP_Create_Post_Notifications:

Diese Prozedur generiert Benachrichtigungen für Benutzer, wenn ein neuer Blog-Post erstellt wird. Dies wird verwendet, um Abonnenten über neue Inhalte zu informieren, die sie interessieren könnten.

SP_Backup_All_DB:

Diese Prozedur erstellt ein Backup aller Datenbanken auf dem Server. Sie wird verwendet, um regelmäßig Sicherungskopien aller Datenbanken zu erstellen, um Datenverluste zu vermeiden.

SP_Backup_Database:

Diese Prozedur erstellt ein Backup einer spezifischen Datenbank. Sie wird verwendet, um eine Sicherungskopie einer einzelnen Datenbank zu erstellen, die bei Bedarf wiederhergestellt werden kann.

Trigger (TR)

TR_BlogPosts_Update_Log:

Dieser Trigger überwacht und protokolliert alle Aktualisierungen in der **BlogPosts**-Tabelle. Er protokolliert Updates an den Feldern **post_title**, **post_content**, **is_premium** und **category_id**. Wenn eines dieser Felder aktualisiert wird, schreibt der Trigger auch das aktuelle Datum und die Uhrzeit in die **updated_at**-Spalte des betroffenen **BlogPosts**. Gleichzeitig werden die Änderungen zusammen mit dem Zeitpunkt und dem Benutzer, der die Änderungen vorgenommen hat, in der **PostUpdateLog**-Tabelle gespeichert. Dies ist nützlich, um Änderungen nachzuverfolgen und sicherzustellen, dass alle Updates ordnungsgemäß dokumentiert sind.

TR_BlogPosts_Delete_Log:

Dieser Trigger markiert einen Blog-Post als gelöscht (**is_deleted = 1**), anstatt ihn vollständig aus der Datenbank zu entfernen. Zusätzlich wird ein Log-Eintrag in der **PostUpdateLog**-Tabelle erstellt, der Informationen über den gelöschten Post, den Zeitpunkt der Löschung und den Benutzer, der die Löschung durchgeführt hat, speichert. Dies ermöglicht es, gelöschte Beiträge weiterhin nachzuverfolgen und eine Historie der Löschvorgänge beizubehalten.

TR_BlogPosts_Insert_Log:

Dieser Trigger protokolliert alle Einfügungen in der **BlogPosts**-Tabelle. Sobald ein neuer Blog-Post erstellt wird, werden die Details des neuen Beitrags zusammen mit dem Erstellungszeitpunkt und dem Benutzer, der den Beitrag erstellt hat, in einer Log-Tabelle gespeichert. Dies hilft bei der Überwachung von neuen Inhalten und der Verfolgung von Aktivitäten in der Plattform.

TR_Notify_Subscribers_On_New_Post:

Dieser Trigger wird automatisch ausgelöst, wenn ein neuer Blog-Post erstellt wird. Er verwendet die gespeicherte Prozedur **SP_Create_Post_Notifications**, um Benachrichtigungen für alle Abonnenten zu generieren. Diese Benachrichtigungen informieren die Abonnenten über den neuen Inhalt, um sicherzustellen, dass sie stets auf dem neuesten Stand der veröffentlichten Beiträge sind. Dies stärkt die Bindung der Abonnenten an die Plattform und fördert ihre regelmäßige Interaktion.

Erweiterungsmöglichkeit

Top-Rated Posts:
Die Datenbank könnte erweitert werden, um die "Top-Rated Posts" zu identifizieren und anzuzeigen. Dies könnte auf den durchschnittlichen Sternebewertungen basieren und den Autoren dabei helfen, erfolgreiche Inhalte zu erkennen und hervorzuheben.

Benachrichtigungen für Autoren:
Autoren könnten benachrichtigt werden, wenn ihre Posts eine bestimmte Anzahl von Bewertungen oder Rezensionen erreicht haben. Dies würde die Interaktion zwischen den Autoren und ihren Lesern fördern.

Kategorien-Abonnement:
Benutzer könnten die Möglichkeit erhalten, bestimmte Kategorien zu abonnieren. Dadurch würden sie benachrichtigt, wenn ein neuer Blog-Post in einer von ihnen abonnierten Kategorie veröffentlicht wird. Dies würde die Benutzererfahrung verbessern, indem es ihnen ermöglicht, gezielt über ihre Interessen informiert zu bleiben.

Erweiterte Benutzereinstellungen:
Benutzer könnten ihre Abonnements und Benachrichtigungspräferenzen in ihren Profileinstellungen verwalten. Sie könnten Kategorien hinzufügen oder entfernen und entscheiden, ob sie Benachrichtigungen sofort, täglich oder wöchentlich erhalten möchten.

Analyse der beliebtesten Kategorien:
Die Plattform könnte erweitert werden, um zu analysieren, welche Kategorien die meisten Abonnenten haben. Dies könnte den Autoren helfen, Inhalte zu erstellen, die auf die Interessen der größten Benutzergruppen abgestimmt sind.

Trending Posts:
Einführung einer Funktion, die dem Benutzer die aktuell am meisten gelesenen oder gelikten Posts zeigt. Dies könnte auf der Startseite prominent platziert werden, um Benutzer zu den beliebtesten Inhalten zu führen.

Monatliche/Jährliche Archive:
Blog-Posts könnten in Archive nach Monaten und Jahren sortiert werden, sodass Benutzer leicht auf ältere Inhalte zugreifen können.

Archivsuche:
Ein erweitertes Suchfeld könnte es den Benutzern ermöglichen, spezifische Posts innerhalb eines bestimmten Zeitraums zu suchen.

Einschränkungen für nicht registrierte Benutzer:
Zugang zu Archiven könnte als exklusive Funktion für registrierte Benutzer oder VIP-Mitglieder angeboten werden, was einen zusätzlichen Anreiz zur Registrierung bietet.

Punktesystem für Interaktionen:
Benutzer könnten Punkte sammeln, wenn sie einen Blog-Post lesen, kommentieren, liken oder teilen. Diese Punkte könnten in einer Rangliste sichtbar sein.

Belohnungen für Aktivität:
Erreicht ein Benutzer bestimmte Punkteschwellen, könnten Belohnungen freigeschaltet werden, wie z.B. ein spezieller Badge, ein kostenloser Monat VIP-Status oder die Möglichkeit, als „Top Contributor“ hervorgehoben zu werden.