YouTubeMusic

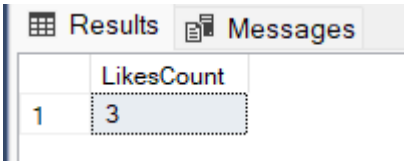
Дигитална трансформация на училищата в България

1. **Резултати от създадените функции, приложени върху данните от база данни YouTubeMusicDB:**

   - **Функция GetSongLikes - функция за броене на лайковете на песен;**

```
CREATE FUNCTION GetSongLikes(@song_id INT)
RETURNS INT
AS
BEGIN
   DECLARE @likes INT;
   SELECT @likes = COUNT(*) FROM UserLikesSong WHERE
song_id = @song_id;
   RETURN @likes;
END;


SELECT dbo.GetSongLikes(50) AS LikesCount;
```

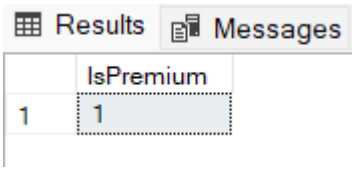


Фигура 1. Резултат от функцията GetSongLikes

   - **Функция IsPremiumUser - функция за проверка дали даден потребител има Premium абонамент;**

```
CREATE FUNCTION IsPremiumUser(@user_id INT)
RETURNS BIT
AS
BEGIN
   DECLARE @result BIT = 0;
   IF EXISTS (
      SELECT 1 FROM Subscription
      WHERE user_id = @user_id AND type = 'Premium'
   )
      SET @result = 1;
   RETURN @result;
END;
```

```
SELECT dbo.IsPremiumUser(5) AS IsPremium;
```



	IsPremium
1	1

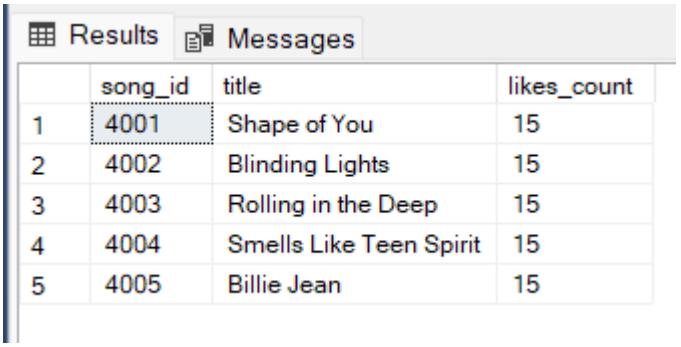
Фигура 2. Резултат от IsPremiumUser.

2. **Резултати от създадените съхранени процедури, приложени върху данните от база данни YouTubeMusicDB:**

   - **Съхранена процедура TopLikedSongs - процедура за намиране 5 TopLikedSongs;**

```
CREATE OR ALTER PROCEDURE TopLikedSongs
AS
BEGIN
   SELECT TOP 5
        s.song_id,
        s.title,
        COUNT(ul.like_id) AS likes_count
   FROM Song s
   JOIN UserLikesSong ul ON s.song_id = ul.song_id
   GROUP BY s.song_id, s.title
   ORDER BY likes_count DESC, s.song_id ASC;
END;


EXEC TopLikedSongs;
```



	song_id	title	likes_count
1	4001	Shape of You	15
2	4002	Blinding Lights	15
3	4003	Rolling in the Deep	15
4	4004	Smells Like Teen Spirit	15
5	4005	Billie Jean	15

Фигура 3. Резултат от съхранената процедура TopLikedSongs.

- **Съхранена процедура GetPlaylistSongs - Процедура за връщане на всички песни в даден плейлист;**

```
CREATE PROCEDURE
GetPlaylistSongs
   @playlist_id INT
AS
BEGIN
   SELECT s.song_id, s.title, s.duration,
s.release_date
   FROM PlaylistItem pi
   JOIN Song s ON pi.song_id =
s.song_id
   WHERE pi.playlist_id = @playlist_id
   ORDER BY pi.position_number;
END;


EXEC GetPlaylistSongs @playlist_id = 5;
```

Results Messages

	song_id	title	duration	release_date
1	31	Magic	260	2014-03-03
2	32	Mockingbird	250	2004-04-25
3	33	Whenever, Wherever	220	2001-08-27
4	34	Hello	300	2015-10-23
5	35	Marvins Room	320	2011-06-09
6	36	Crazy in Love	230	2003-05-14
7	37	Sing	230	2014-04-07
8	38	I Knew You Were Trouble	230	2012-10-09
9	39	Only Girl (In the World)	240	2010-09-10
10	40	Grenade	230	2010-09-28

Фигура 4. Резултат от съхранената процедура GetPlaylistSongs.

3. **Резултати от създадените тригери, приложени върху данните от база данни YouTubeMusicDB:**

- **Тригер trg_SetLikeDate - тригър за автоматично задаване на дата при нов лайк.**

```
CREATE TRIGGER trg_SetLikeDate
ON UserLikesSong
AFTER INSERT
AS
BEGIN
   UPDATE UserLikesSong
   SET liked_at = GETDATE()
   WHERE like_id IN (SELECT like_id FROM inserted);
END;
```

Вмъкваме нов лайк без да задаваме дата:

```
INSERT INTO UserLikesSong (like_id, user_id, song_id)
VALUES (201, 1, 10);
```

Проверяваме дали тригерът е попълнил датата:

```
SELECT like_id, user_id, song_id, liked_at
FROM UserLikesSong
WHERE like_id = 201;
```

Фигура 5. Резултат от тригър trg_SetLikeDate.

- **Тригер trg_SetCommentDate- тригър за автоматично задаване на текуща дата при добавяне на нов коментар**

```
CREATE TRIGGER trg_SetCommentDate
ON Comment
AFTER INSERT
AS
BEGIN
   UPDATE Comment
   SET posted_at = GETDATE()
   WHERE comment_id IN (SELECT comment_id FROM inserted);
END;
```

Вмъкваме нов коментар без да задаваме дата

```
INSERT INTO Comment (comment_id, content, user_id, song_id)
VALUES (101, 'Test trigger comment', 2, 15);
```

Проверяваме дали тригерът е попълнил датата
SELECT comment_id, content, posted_at, user_id, song_id
FROM Comment
WHERE comment_id = 101;

Results Messages

	comment_id	content	posted_at	user_id	song_id
1	101	Test trigger comment	2025-11-29 00:36:35.337	2	15

Фигура 5. Резултат от тригър trg_SetCommentDate.