Advanced transactional queries:

```
-- unify approved user submitted questions from the UserSubmission table with
-- official questions from the Question table on any category
-- returned table is what should be queried on for standard mode question pool
-- advanced through set operator and subquery
create view CombinedQuestions as
select
'Official' as QuestionSource,
q.QuestionID as ID,
q.QuestionText,
q.CorrectAnswer,
q.IncorrectAns1,
q.IncorrectAns2,
q.IncorrectAns3,
q.Difficulty,
q.CategoryID
from Question q
join Category c on q.CategoryID = c.CategoryID

union all

select
'UserSubmitted' as QuestionSource,
us.SubmissionID as ID,
us.QuestionText,
us.CorrectAnswer,
us.IncorrectAns1,
us.IncorrectAns2,
us.IncorrectAns3,
us.Difficulty,
us.CategoryID
from UserSubmission us
join Category c on us.CategoryID = c.CategoryID
where us.Status = 1;

-- playing standard mode on NFL category
-- advanced through set operators and subqueries
(select * from CombinedQuestions
where CategoryID in (
```

```
    select CategoryID from Category where Type = 'NFL'
) and Difficulty = 1
order by rand()
limit 4)

union all

(select * from CombinedQuestions
where CategoryID in (
    select CategoryID from Category where Type = 'NFL'
) and Difficulty = 2
order by rand()
limit 3)

union all

(select * from CombinedQuestions
where CategoryID in (
    select CategoryID from Category where Type = 'NFL'
) and Difficulty = 3
order by rand()
limit 3);


-- playing standard mode on CFB category
-- advanced through set operators and subqueries
select * from CombinedQuestions
where CategoryID = (
    select CategoryID from Category where Type = 'CFB'
) and Difficulty = 1
order by rand()
limit 4

union all

select * from CombinedQuestions
where CategoryID = (
    select CategoryID from Category where Type = 'CFB'
) and Difficulty = 2
order by rand()
```

```
limit 3

union all

select * from CombinedQuestions
where CategoryID = (
    select CategoryID from Category where Type = 'CFB'
) and Difficulty = 3
order by rand()
limit 3;

-- top 10 players' usernames and high scores for NFL on standard mode
-- not advanced
select u.Username, uhs.HighScore
from UserHighScore uhs
inner join User u on uhs.UserID = u.UserID
inner join Category c on uhs.CategoryID = c.CategoryID
where c.Type = 'NFL' and uhs.TriviaMode = 'Standard'
order by uhs.HighScore desc
limit 10;
```

Trigger: Ensuring user submitted question are correctly formatted

```
DELIMITER //
CREATE TRIGGER check_question_format
BEFORE INSERT ON UserSubmission
FOR EACH ROW
BEGIN
  -- Basic formatting check
        IF CHAR_LENGTH(NEW.QuestionText) < 10
                OR RIGHT(NEW.QuestionText, 1) != '?' THEN

                -- Set status to 0 (failed formatting) instead of rejecting insert
                SET NEW.Status = 0;
        ELSE
                -- If it passes, set status to 1 (ok)
                SET NEW.Status = 1;
        END IF;
END;
//
```

```
DELIMITER ;
```

Store Procedure

– Query for updating high score table after a user beats their score for a category

```
Create Procedure UpdateUserHighScore(
    In updated_UserID Int,
    In updated_CategoryID Int,
    In updated_TriviaMode Varchar(10),
    In updated_NewScore Int
)
Begin
    Declare oldScore Int;
    Declare userName Varchar(255);

    Select uh.HighScore, u.Username
    Into oldScore, userName
    From UserHighScore uh
    Join User u On uh.UserID = u.UserID
    Where uh.UserID = updated_UserID
      And uh.CategoryID = updated_CategoryID
      And uh.TriviaMode = updated_TriviaMode
    Limit 1;
```

```
    If oldScore Is Null Then
        Insert Into UserHighScore (UserID, CategoryID, TriviaMode, HighScore)
        Values (updated_UserID, updated_CategoryID, updated_TriviaMode, updated_NewScore);
    Elseif updated_NewScore > oldScore Then
        Update UserHighScore
        Set HighScore = updated_NewScore
        Where UserID = updated_UserID
          And CategoryID = updated_CategoryID
          And TriviaMode = updated_TriviaMode;
    End If;
End;
```

– User creation updates both User database as well as HighScore database

```
Create Procedure CreateNewUser(
    In new_Username Varchar(255),
    In new_Email Varchar(255),
    In new_PasswordHash Varchar(255),
    In new_RegistrationDate Date
)
Begin
    Declare new_UserID Int;
    Declare nfl_CategoryID Int;
    Declare cfb_CategoryID Int;

    Select Max(UserID) Into new_UserID From User;
    Set new_UserID = new_UserID + 1;


    Insert Ignore Into User (UserID, Username, Email, PasswordHash, RegistrationDate)
    Values (new_UserID, new_Username, new_Email, new_PasswordHash,
new_RegistrationDate);
```

```
    Select c.CategoryID
    Into nfl_CategoryID
    From Category c
    Where c.Type = 'NFL'
      And c.CategoryID = (Select Min(CategoryID) From Category Where Type = 'NFL');

    Insert Ignore Into UserHighScore (UserID, CategoryID, TriviaMode, HighScore)
    Values (new_UserID, nfl_CategoryID, 'Standard', 0),
           (new_UserID, nfl_CategoryID, 'Hard', 0);

    Select c.CategoryID
    Into cfb_CategoryID
    From Category c
    Where c.Type = 'CFB'
      And c.CategoryID = (Select Min(CategoryID) From Category Where Type = 'CFB');

    Insert Ignore Into UserHighScore (UserID, CategoryID, TriviaMode, HighScore)
    Values (new_UserID, cfb_CategoryID, 'Standard', 0),
           (new_UserID, cfb_CategoryID, 'Hard', 0);
End;



SELECT * FROM Question WHERE QuestionText LIKE '%{submission}%';

End;
```