Stored Procedure:

```
DELIMITER //

CREATE PROCEDURE BasicPlayerStats()
BEGIN
    DECLARE done INT DEFAULT FALSE;
    DECLARE playerId VARCHAR(255);
    DECLARE playerName VARCHAR(255);
    DECLARE totalScore INT;
    DECLARE playerTag VARCHAR(255);
    DECLARE matchesPlayed INT;

    DECLARE topScorers CURSOR FOR
    SELECT p.player_id, p.player_name, SUM(g.core_goals) AS total_score
    FROM PlayerK p JOIN games_by_players g ON p.player_id = g.player_id
    GROUP BY p.player_id, p.player_name
    ORDER BY total_score DESC;

    DECLARE matchCounts CURSOR FOR
    SELECT m.player_tag, COUNT(g.game_id) AS matches_played
    FROM GameK g JOIN MatchK m ON g.player_tag = m.player_tag
    GROUP BY m.player_tag
    ORDER BY matches_played DESC;

    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

    DROP TEMPORARY TABLE IF EXISTS PlayerStats;
    CREATE TEMPORARY TABLE PlayerStats (
        player_id VARCHAR(255),
        player_name VARCHAR(255),
        total_score INT,
        matches_played INT
    );

    OPEN topScorers;
    FETCH topScorers INTO playerId, playerName, totalScore;
    WHILE NOT done DO
        INSERT INTO PlayerStats (player_id, player_name, total_score,
matches_played)
```

```
        VALUES (playerId, playerName, totalScore, NULL);
        FETCH topScorers INTO playerId, playerName, totalScore;
    END WHILE;
    CLOSE topScorers;

    SET done = FALSE;

    OPEN matchCounts;
    FETCH matchCounts INTO playerTag, matchesPlayed;
    WHILE NOT done DO
        UPDATE PlayerStats
        SET matches_played = matchesPlayed
        WHERE player_id = (SELECT player_id FROM PlayerK WHERE player_tag
= playerTag);
        FETCH matchCounts INTO playerTag, matchesPlayed;
    END WHILE;
    CLOSE matchCounts;

    SELECT * FROM PlayerStats;

    DROP TEMPORARY TABLE IF EXISTS PlayerStats;
END //

DELIMITER ;
```

Constraint:

- In addition to primary keys and foreign keys

```
CREATE ASSERTION LessTeams CHECK (
    (SELECT COUNT(*) FROM Team) <= (SELECT COUNT(*) FROM Player)
);
```

Trigger:

```
DELIMITER //

CREATE TRIGGER rejectPlayer
BEFORE INSERT ON PlayerK
FOR EACH ROW
BEGIN
    DECLARE cnt INT;
```

```
    SELECT COUNT(*) INTO cnt
    FROM PlayerK
    WHERE player_id = NEW.player_id;

    IF cnt > 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Duplicate player_id is
not allowed';
    END IF;
END //

DELIMITER ;
```

Transaction:

```
SET SESSION TRANSACTION ISOLATION LEVEL READ COMMITTED;

START TRANSACTION;

DROP TEMPORARY TABLE IF EXISTS TempPlayerScores;
CREATE TEMPORARY TABLE TempPlayerScores (
    player_id VARCHAR(255),
    player_name VARCHAR(255),
    total_score INT
);

DROP TEMPORARY TABLE IF EXISTS TempMatchCounts;
CREATE TEMPORARY TABLE TempMatchCounts (
    player_tag VARCHAR(255),
    matches_played INT
);

INSERT INTO TempPlayerScores (player_id, player_name, total_score)
SELECT p.player_id, p.player_name, SUM(g.core_goals) AS total_score
FROM PlayerK p
JOIN games_by_players g ON p.player_id = g.player_id
GROUP BY p.player_id, p.player_name
ORDER BY total_score DESC;
```

```sql
INSERT INTO TempMatchCounts (player_tag, matches_played)
SELECT m.player_tag, COUNT(g.game_id) AS matches_played
FROM GameK g
JOIN MatchK m ON g.player_tag = m.player_tag
GROUP BY m.player_tag
ORDER BY matches_played DESC;

COMMIT;

SELECT * FROM TempPlayerScores;
SELECT * FROM TempMatchCounts;

DROP TEMPORARY TABLE IF EXISTS TempPlayerScores;
DROP TEMPORARY TABLE IF EXISTS TempMatchCounts;
```