Basketball Player Performance Database

Goal: Store player stats and analyze performance trends.

Tables:

- `players(player_id, name, team, position)`
- `games(game_id, date, opponent)`
- `stats(player_id, game_id, points, rebounds, assists, steals, blocks)`

Queries:

- Find average points per player.
- Identify top rebounders in a season.
- Compare team performance across opponents.

a. Create database

```
CREATE DATABASE basketball_performance;
USE basketball_performance;
```

```
mysql> CREATE DATABASE basketball_performance;
Query OK, 1 row affected (0.29 sec)

mysql> USE basketball_performance;
Database changed
```

b. Create tables

```
-- Player information
CREATE TABLE players (
    player_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    team VARCHAR(50),
    position VARCHAR(20)
);

-- Game information
CREATE TABLE games (
    game_id INT AUTO_INCREMENT PRIMARY KEY,
    date DATE,
    opponent VARCHAR(50)
);

-- Player stats per game
CREATE TABLE stats (
    stat_id INT AUTO_INCREMENT PRIMARY KEY,
    player_id INT,
    game_id INT,
    points INT,
    rebounds INT,
    assists INT,
    steals INT,
    blocks INT,
    FOREIGN KEY (player_id) REFERENCES players(player_id),
    FOREIGN KEY (game_id) REFERENCES games(game_id)
);
```

```
mysql> -- Player information
mysql> CREATE TABLE players (
    ->     player_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     team VARCHAR(50),
    ->     position VARCHAR(20)
    -> );
Query OK, 0 rows affected (0.20 sec)

mysql>
mysql> -- Game information
mysql> CREATE TABLE games (
    ->     game_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     date DATE,
    ->     opponent VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.11 sec)

mysql>
mysql> -- Player stats per game
mysql> CREATE TABLE stats (
    ->     stat_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     player_id INT,
    ->     game_id INT,
    ->     points INT,
    ->     rebounds INT,
    ->     assists INT,
    ->     steals INT,
    ->     blocks INT,
    ->     FOREIGN KEY (player_id) REFERENCES players(player_id),
    ->     FOREIGN KEY (game_id) REFERENCES games(game_id)
    -> );
Query OK, 0 rows affected (0.34 sec)
```

c. Insert sample data

```
INSERT INTO players (name, team, position)
VALUES
('Alice Johnson', 'Jakarta Tigers', 'Guard'),
('Bob Smith', 'Jakarta Tigers', 'Forward'),
('Charlie Lee', 'Bandung Eagles', 'Center');

INSERT INTO games (date, opponent)
VALUES
('2025-11-01', 'Bandung Eagles'),
('2025-11-05', 'Surabaya Sharks');

INSERT INTO stats (player_id, game_id, points, rebounds, assists,
steals, blocks)
VALUES
(1, 1, 25, 5, 7, 2, 0),
(2, 1, 18, 10, 3, 1, 2),
(3, 1, 12, 12, 2, 0, 3),
(1, 2, 30, 4, 8, 3, 1),
(2, 2, 20, 9, 4, 2, 1);
```

```
mysql> INSERT INTO players (name, team, position)
    -> VALUES
    -> ('Alice Johnson', 'Jakarta Tigers', 'Guard'),
    -> ('Bob Smith', 'Jakarta Tigers', 'Forward'),
    -> ('Charlie Lee', 'Bandung Eagles', 'Center');
Query OK, 3 rows affected (0.03 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO games (date, opponent)
    -> VALUES
    -> ('2025-11-01', 'Bandung Eagles'),
    -> ('2025-11-05', 'Surabaya Sharks');
Query OK, 2 rows affected (0.03 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO stats (player_id, game_id, points, rebounds, assists, steals, blocks)
    -> VALUES
    -> (1, 1, 25, 5, 7, 2, 0),
    -> (2, 1, 18, 10, 3, 1, 2),
    -> (3, 1, 12, 12, 2, 0, 3),
    -> (1, 2, 30, 4, 8, 3, 1),
    -> (2, 2, 20, 9, 4, 2, 1);
Query OK, 5 rows affected (0.03 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

d. Example queries

```
-- Average points per player
SELECT p.name, AVG(s.points) AS avg_points
FROM players p
JOIN stats s ON p.player_id = s.player_id
GROUP BY p.name;

-- Top rebounders in a season
SELECT p.name, SUM(s.rebounds) AS total_rebounds
FROM players p
JOIN stats s ON p.player_id = s.player_id
GROUP BY p.name
ORDER BY total_rebounds DESC;

-- Team performance against opponents
SELECT p.team, g.opponent, SUM(s.points) AS team_points
FROM players p
JOIN stats s ON p.player_id = s.player_id
JOIN games g ON s.game_id = g.game_id
GROUP BY p.team, g.opponent;
```

```
mysql> -- Average points per player
mysql> SELECT p.name, AVG(s.points) AS avg_points
    -> FROM players p
    -> JOIN stats s ON p.player_id = s.player_id
    -> GROUP BY p.name;
+--------------+------------+
| name         | avg_points |
+--------------+------------+
| Alice Johnson |    27.5000 |
| Bob Smith     |    19.0000 |
| Charlie Lee   |    12.0000 |
+--------------+------------+
3 rows in set (0.05 sec)

mysql>
mysql> -- Top rebounders in a season
mysql> SELECT p.name, SUM(s.rebounds) AS total_rebounds
    -> FROM players p
    -> JOIN stats s ON p.player_id = s.player_id
    -> GROUP BY p.name
    -> ORDER BY total_rebounds DESC;
+--------------+----------------+
| name         | total_rebounds |
+--------------+----------------+
| Bob Smith     |             19 |
| Charlie Lee   |             12 |
| Alice Johnson |              9 |
+--------------+----------------+
3 rows in set (0.00 sec)
```

```
mysql>
mysql> -- Team performance against opponents
mysql> SELECT p.team, g.opponent, SUM(s.points) AS team_points
    -> FROM players p
    -> JOIN stats s ON p.player_id = s.player_id
    -> JOIN games g ON s.game_id = g.game_id
    -> GROUP BY p.team, g.opponent;
+----------------+----------------+-------------+
| team           | opponent       | team_points |
+----------------+----------------+-------------+
| Jakarta Tigers | Bandung Eagles |          43 |
| Bandung Eagles | Bandung Eagles |          12 |
| Jakarta Tigers | Surabaya Sharks |          50 |
+----------------+----------------+-------------+
3 rows in set (0.00 sec)

mysql>
```