School of Computer Science and Engineering

CSE2004

IPL DATABASE USING MYSQL

NAME	REGISTRATION NUMBER
Vaasu Gupta	16BCB0062
Aryan Soni	16BCI0198
Akshchat Arya	16BCE0426

CERTIFICATE

This is to certify that the project work entitled “IPL Database” that is being submitted by “Aryan Soni, Akshchat Arya and Vaasu Gupta” for Database Management Systems (CSE2004) is a record of bonafide work done under my supervision. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted for any other CAL course.

Place: Vellore

Date:

Signature of Faculty:

Signature of Students:

16BCB0062 –Vaasu Gupta

16BCI0198-Aryan Soni

16BCE0426-Akshchat Arya

ACKNOWLEDGEMENT

We would like to thank VIT University for providing us an opportunity to carry out this research project. We also thank to our batch mates and seniors in helping us to carry out our project work.

We thank Prof. SHARMILA BANU to be our project guide and guided us at various stages in completing the project. Without Her support it would be very difficult for us to do the project. We would like to pay our gratitude to Maa’m for sharing his pearls of wisdom with us during the project.

INTRODUCTION

Why Database management System is important in Real world problems?

1>Controlling Data Redundancy

In non-database frameworks every application program has its own private documents. For this situation, the copied duplicates of similar information is made in many spots. In DBMS, all information of an association is incorporated into a solitary database document. The information is recorded in just a single place in the database and it is not copied.

2>Sharing of Data

In DBMS, information can be shared by approved clients of the association. The database head deals with the information and offers rights to clients to get to the information. Numerous clients can be approved to get to a similar snippet of data at the same time. The remote clients can likewise share same information. Essentially, the information of same database can be shared between various application programs.

3>Data Consistency

By controlling the information excess, the information consistency is acquired. In the event that an information thing seems just once, any refresh to its esteem must be performed just once and the refreshed esteem is instantly accessible to all clients. In the event that the DBMS has controlled excess, the database framework upholds consistency.

4>Data Independence

The division of information structure of database from the application program that uses the information is called information autonomy. In DBMS, you can undoubtedly change the structure of database without adjusting the application program.

5>Integration constraints

The division of information structure of database from the application program that uses the information is called data independence. In DBMS, you can undoubtedly change the structure of database without adjusting the application program.

And many more…..

About Indian Premier league:

The **Indian Premier League** is a professional Twenty20 cricket league in India contested during April and May of every year by teams representing Indian cities. The league was founded by the Board of Control for Cricket in India (BCCI)

Why database management system is necessary for IPL?

As we all know that the IPL is one of the biggest sporting event in India as well as world hence there is a great need for organising and maintaining several records regarding this game to ensure the 'smooth management of the game'.

There will be several areas that needs to be covered in this project:

1> Player details
2> Team and it's member's details
3> Match details
4> Player performance details
5> Fixture
6> Player Stats

Bulk Insert

A Bulk insert is a process or method provided by a database management system to load multiple rows of data into a database table.

Bulk insert may refer to:

Transact-SQL BULK INSERT statement

PL/SQL BULK COLLECT and FORALL statements

MySQL LOAD DATA INFILE statement

We are able to bulk insert data either from a csv file or mysql shell.

Attribute Constraints

Attributes can have constraints associated with them, which typically indicate such things as maximum value, minimum value and length of field. You define these constraints within the attribute properties.

Referential Integrity Constraints

Referential integrity (RI) is a relational database concept, which states that table relationships must always be consistent. In other words, any foreign key field must agree with the primary key that is referenced by the foreign key.

Foreign key constraints:

Foreign Key: fk_coach_has_player_coach1

Definition:

Target	coach (coach_id1 → coach_id)
On Update	NO ACTION
On Delete	NO ACTION

Foreign Key: fk_coach_has_player_player1

Definition:

Target	player (player_id1 → player_id)
On Update	NO ACTION
On Delete	NO ACTION

Foreign Key: fk_fixture_stadium1

Definition:

Target	stadium (stadium>stadium_id → stadium_id)
On Update	NO ACTION
On Delete	NO ACTION

Foreign Key: fk1

Definition:

Target	player (player_id → player_id)
On Update	NO ACTION
On Delete	NO ACTION

Foreign Key: fk2

Definition:

Target	fixture (fixture_id → fixture_id)
On Update	NO ACTION
On Delete	NO ACTION

Foreign Key: fk3

Definition:

Target	stadium (stadium>stadium_id → stadium_id)
On Update	NO ACTION
On Delete	NO ACTION

Foreign Key: fk4

Definition:

Target	team (team>team_id → team_id)
On Update	NO ACTION
On Delete	NO ACTION

Datatypes used in our database:

VARCHAR

INT

DATE

UNSIGNED INT

TIME

Query for database creation:

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS,
UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;

SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

CREATE SCHEMA IF NOT EXISTS `ubuntu` DEFAULT CHARACTER SET
latin1 ;

USE `ubuntu` ;

CREATE TABLE IF NOT EXISTS `ubuntu`.`coach` (

  `coach_id` INT(11) NOT NULL,

  `name` VARCHAR(45) NOT NULL,

  `nationality` VARCHAR(45) NOT NULL,

  `speciality` VARCHAR(45) NULL DEFAULT NULL,

  PRIMARY KEY (`coach_id`))

  row_format=compressed

  key_block_size=8

ENGINE = InnoDB

DEFAULT CHARACTER SET = latin1;
```

```
CREATE TABLE IF NOT EXISTS `ubuntu`.`player` (
  `player_id` INT(11) NOT NULL,
  `player_name` VARCHAR(45) NOT NULL,
  `nationalty` VARCHAR(45) NOT NULL,
  `team_name` VARCHAR(45) NOT NULL,
  `age` INT(11) NOT NULL,
  `specialization` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`player_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;


CREATE TABLE IF NOT EXISTS `ubuntu`.`coach_has_player` (
  `coach>coach_id` INT(11) NOT NULL,
  `player>player_id` INT(11) NOT NULL,
  PRIMARY KEY (`coach>coach_id`, `player>player_id`),
  INDEX `fk_coach_has_player_player1_idx` (`player>player_id` ASC),
  INDEX `fk_coach_has_player_coach1_idx` (`coach>coach_id` ASC),
  CONSTRAINT `fk_coach_has_player_coach1`
    FOREIGN KEY (`coach>coach_id`)
    REFERENCES `ubuntu`.`coach` (`coach_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_coach_has_player_player1`
    FOREIGN KEY (`player>player_id`)
    REFERENCES `ubuntu`.`player` (`player_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;
CREATE TABLE IF NOT EXISTS `ubuntu`.`stadium` (
  `stadium_id` INT(11) NOT NULL,
  `name` VARCHAR(45) NULL DEFAULT NULL,
  `capacity` INT(11) NULL DEFAULT NULL,
  `average_attendance` INT(11) NULL DEFAULT NULL,
  `perimeter` INT(11) NULL DEFAULT NULL,
  PRIMARY KEY (`stadium_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;

CREATE TABLE IF NOT EXISTS `ubuntu`.`fixture` (
  `fixture_id` INT(11) NOT NULL,
  `venue` VARCHAR(45) NOT NULL,
  `result` VARCHAR(45) NOT NULL,
  `date` DATETIME NOT NULL,
  `name` VARCHAR(45) NOT NULL,
  `stadium>stadium_id` INT(11) NOT NULL,
  `winning_team_id` INT(11) NOT NULL,
  PRIMARY KEY (`fixture_id`),
  INDEX `fk_fixture_stadium1_idx` (`stadium>stadium_id` ASC),
  CONSTRAINT `fk_fixture_stadium1`
    FOREIGN KEY (`stadium>stadium_id`)
    REFERENCES `ubuntu`.`stadium` (`stadium_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;


CREATE TABLE IF NOT EXISTS `ubuntu`.`player_performance` (
  `player_id` INT(11) NOT NULL,
  `fixture_id` INT(11) NOT NULL,
  `player_total_runs` INT(11) NULL DEFAULT NULL,
  `6` INT(11) NULL DEFAULT NULL,
  `4` INT(11) NULL DEFAULT NULL,
  `strike_rate` INT(11) NULL DEFAULT NULL,
  `total_wickets` INT(11) NULL DEFAULT NULL,
  `average_speed` VARCHAR(45) NULL DEFAULT NULL,
  `extras` VARCHAR(45) NULL DEFAULT NULL,
  `total_catches` INT(11) NULL DEFAULT NULL,
  `run_Out` INT(11) NULL DEFAULT NULL,
  PRIMARY KEY (`player_id`, `fixture_id`),
  INDEX `fk2_idx` (`fixture_id` ASC),
  CONSTRAINT `fk1`
    FOREIGN KEY (`player_id`)
    REFERENCES `ubuntu`.`player` (`player_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk2`
    FOREIGN KEY (`fixture_id`)
    REFERENCES `ubuntu`.`fixture` (`fixture_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;



CREATE TABLE IF NOT EXISTS `ubuntu`.`team` (
  `team_id` INT(11) NOT NULL,
  `team_name` VARCHAR(45) NOT NULL,
  `owner` VARCHAR(45) NOT NULL,
  `coach>coach_id` INT(11) NOT NULL,
  PRIMARY KEY (`team_id`),
  INDEX `fk_team_coach1_idx` (`coach>coach_id` ASC),
  CONSTRAINT `fk_team_coach1`
```

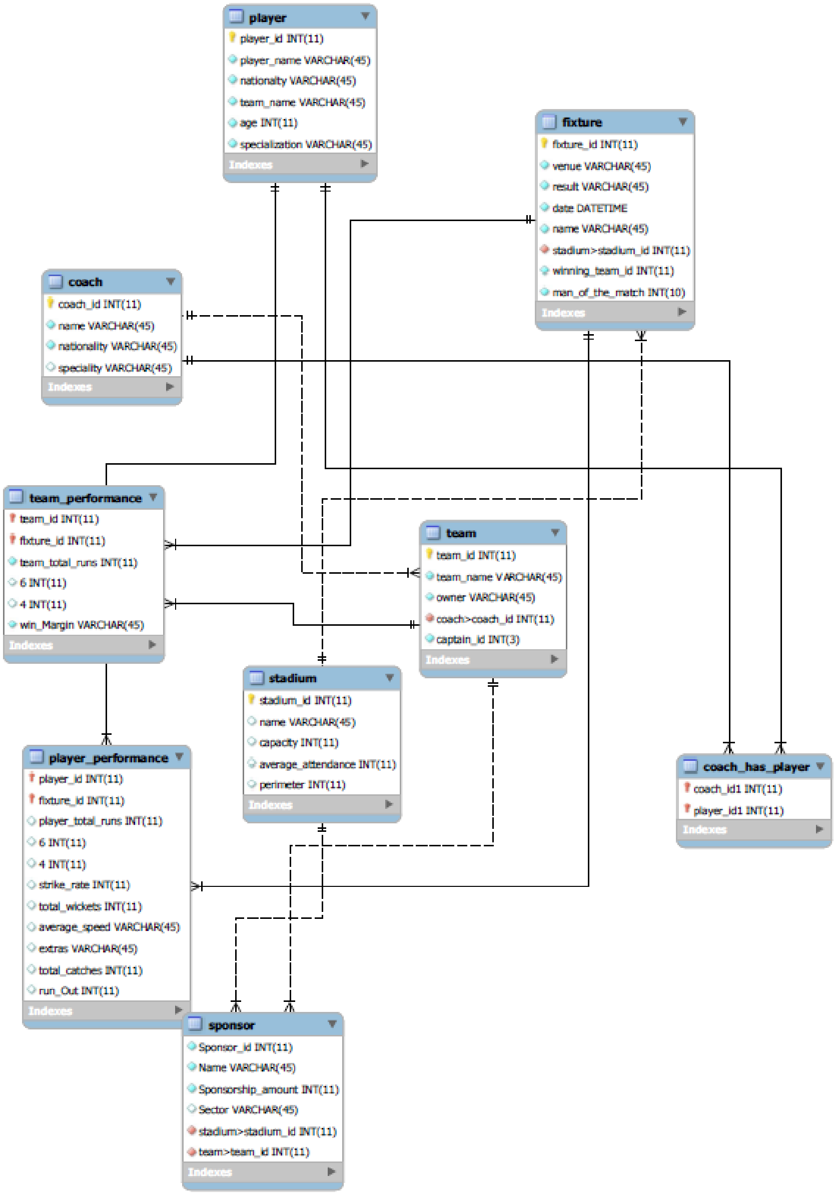
```
    FOREIGN KEY (`coach>coach_id`)
    REFERENCES `ubuntu`.`coach` (`coach_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;

CREATE TABLE IF NOT EXISTS `ubuntu`.`sponsor` (
  `Sponsor_id` INT(11) NOT NULL,
  `Name` VARCHAR(45) NOT NULL,
  `Sponsorship_amount` INT(11) NOT NULL,
  `Sector` VARCHAR(45) NULL DEFAULT NULL,
  `stadium>stadium_id` INT(11) NOT NULL,
  `team>team_id` INT(11) NOT NULL,
  INDEX `fk_sponsor_stadium1_idx` (`stadium>stadium_id` ASC),
  INDEX `fk4_idx` (`team>team_id` ASC),
  CONSTRAINT `fk3`
    FOREIGN KEY (`stadium>stadium_id`)
    REFERENCES `ubuntu`.`stadium` (`stadium_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk4`
    FOREIGN KEY (`team>team_id`)
    REFERENCES `ubuntu`.`team` (`team_id`)
```

```
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;

CREATE TABLE IF NOT EXISTS `ubuntu`.`team_performance` (
  `team_id` INT(11) NOT NULL,
  `fixture_id` INT(11) NOT NULL,
  `team_total_runs` INT(11) NOT NULL,
  `6` INT(11) NULL DEFAULT NULL,
  `4` INT(11) NULL DEFAULT NULL,
  `win_Margin` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`team_id`, `fixture_id`),
  INDEX `fk_team_has_fixture_fixture1_idx` (`fixture_id` ASC),
  INDEX `fk_team_has_fixture_team1_idx` (`team_id` ASC),
  CONSTRAINT `fk_team_has_fixture_fixture1`
    FOREIGN KEY (`fixture_id`)
    REFERENCES `ubuntu`.`fixture` (`fixture_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_team_has_fixture_team1`
    FOREIGN KEY (`team_id`)
    REFERENCES `ubuntu`.`team` (`team_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;



SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

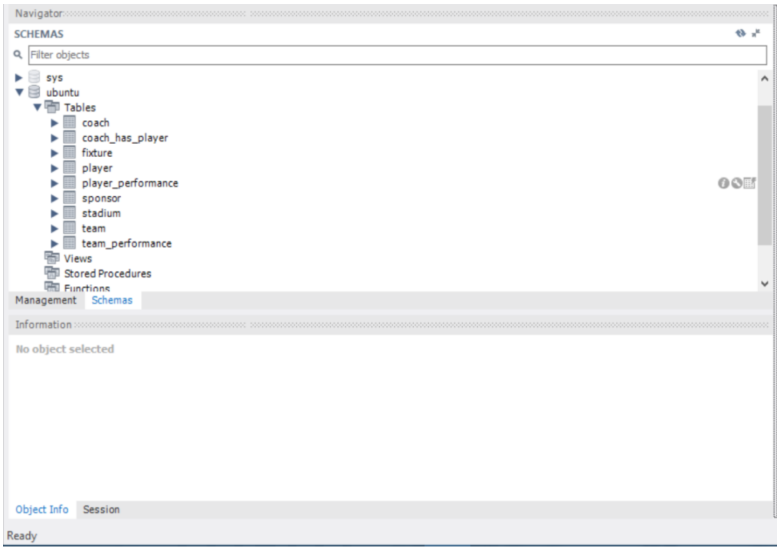
Business Rules For This Schema

- This Schema is made with the rules of the Indian premier league in mind.
- Here the owners each can have only one team situated in one venue (eg RCB is situated in Bangalore).
- There are stadiums in our schema each fixture can be played in one stadium, each venue can have one or stadiums (eg Bangalore has s1 and s2).
- There are sponsors who can each fund one team as well as one stadium. (eg Lenovo funds RCB and s1 each for 10000000 rupees).
- There are players every player can play only for one team and each team can have many players.
- Fixture gives the information about each match the winner, venue etc.
- Each team has to play at least one fixture.
- Each player has its own player_performance and team has team_performance for every fixture they take part in.
- There are coaches who can coach one team but many players, each team has only coach and each player can have one or more coaches.
- Every team has to play a fixture against one another.
- The team that wins the most no of matches wins the tournament.
- If there are two teams with same n.o of points, then there will be an extra match between those two teams to determine the winner.

ER Diagram/s for database



Schema/Tables for data base



Data Compression

Since processors and cache memories have expanded in speed more than circle stockpiling gadgets, numerous workloads are plate bound. Data compression empowers littler database measure, decreased I/O, and enhanced throughput, at the little cost of expanded CPU use. Compression is particularly profitable for read-intensive applications, on frameworks with enough RAM to keep as often as possible utilized data in memory.

An InnoDB table created with ROW_FORMAT=COMPRESSED can use a smaller page size on disk than the usual 16KB default. Smaller pages require less I/O to read from and write to disk, which is especially valuable for SSD devices.

The page size is specified through the KEY_BLOCK_SIZE parameter. The different page size means the table must be in its own .ibd file rather than in the system tablespace, which requires enabling the innodb_file_per_table option. The level of compression is the same regardless of the KEY_BLOCK_SIZE value. As you specify smaller values for KEY_BLOCK_SIZE, you get the I/O benefits of increasingly smaller pages. But if you specify a value that is too small, there is additional overhead to reorganize the pages when data values cannot be compressed enough to fit multiple rows in each page. There is a hard limit on how small KEY_BLOCK_SIZE can be for a table, based on the lengths of the key columns for each of its indexes. Specify a value that is too small, and the CREATE TABLE or ALTER TABLE statement fails.

How will we compress our databse in mysql:

Before making a compressed table, ensure the innodb_file_per_table setup choice is enabled, and innodb_file_format is set to Barracuda. You can set these parameters in the MySQL setup record my.cnf or my.ini, or with the SET explanation without closing down the MySQL server.

To empower compression for a table, you utilize the provisions ROW_FORMAT=COMPRESSED, KEY_BLOCK_SIZE, or both in a CREATE TABLE or ALTER TABLE proclamation.

Example:

```
SET GLOBAL innodb_file_per_table=1;

SET GLOBAL innodb_file_format=Barracuda;

CREATE TABLE t1

 (c1 INT PRIMARY KEY)

 ROW_FORMAT=COMPRESSED

 KEY_BLOCK_SIZE=8;
```

If you specify ROW_FORMAT=COMPRESSED, you can omit KEY_BLOCK_SIZE; the default page size value is used, which is half the innodb_page_size value.

If you specify KEY_BLOCK_SIZE, you can omit ROW_FORMAT=COMPRESSED; compression is enabled automatically.

To determine the best value for KEY_BLOCK_SIZE, typically you create several copies of the same table with different values for this clause, then measure the size of the resulting .ibd files and see how well each performs with a realistic workload.

We will be using compression in database approach:

When enabled, MySQL table compression is automatic and applies to all columns and index values. The columns can still be tested with operators such as LIKE, and sort operations can still use indexes even when the index values are compressed. Because indexes are often a significant fraction of the total size of a database, compression could result in significant savings in storage, I/O or processor time. The compression and decompression operations happen on the database server, which likely is a powerful system that is sized to handle the expected load.

Multi Lingual Data

You can insert any language text in MySQL Table by changing the Collation of the table Field to 'utf8_general_ci

SQL Injection

To Set a database to read-only mode in Mysql:

```
FLUSH TABLES WITH READ LOCK;

SET GLOBAL read_only = 1;
```

To Set the database back to Read+Write mode:

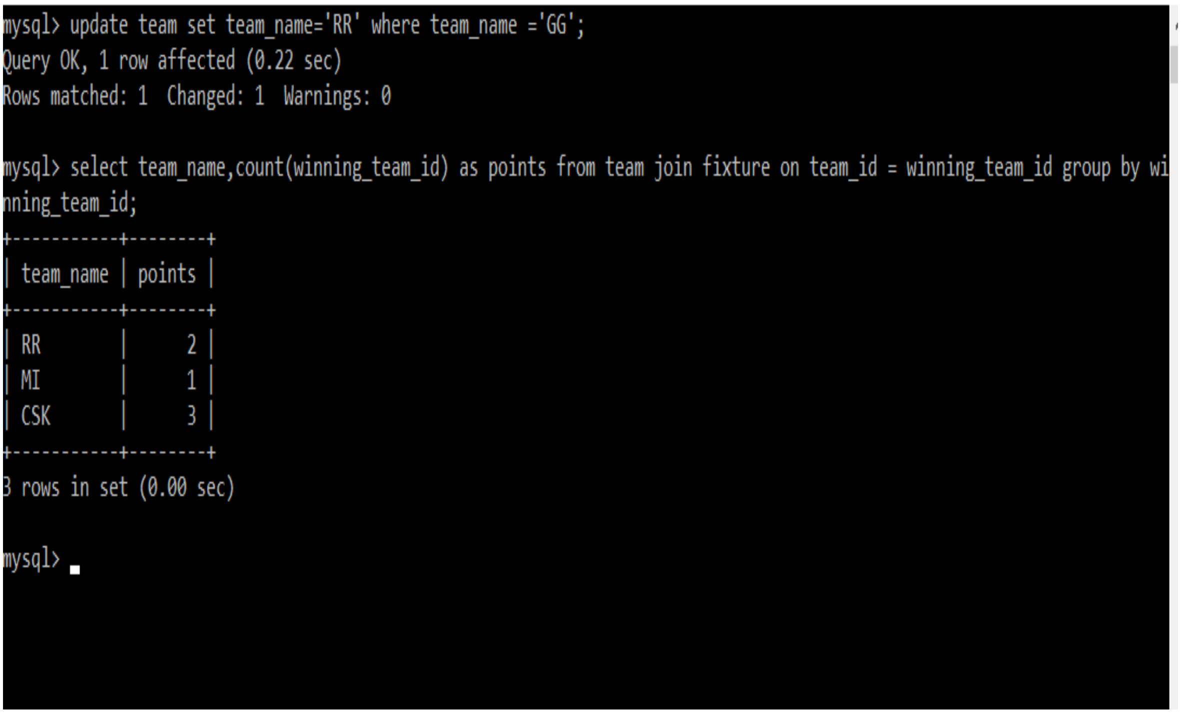
SET GLOBAL read_only = 0;

UNLOCK TABLES;

Our Review 2 Queries

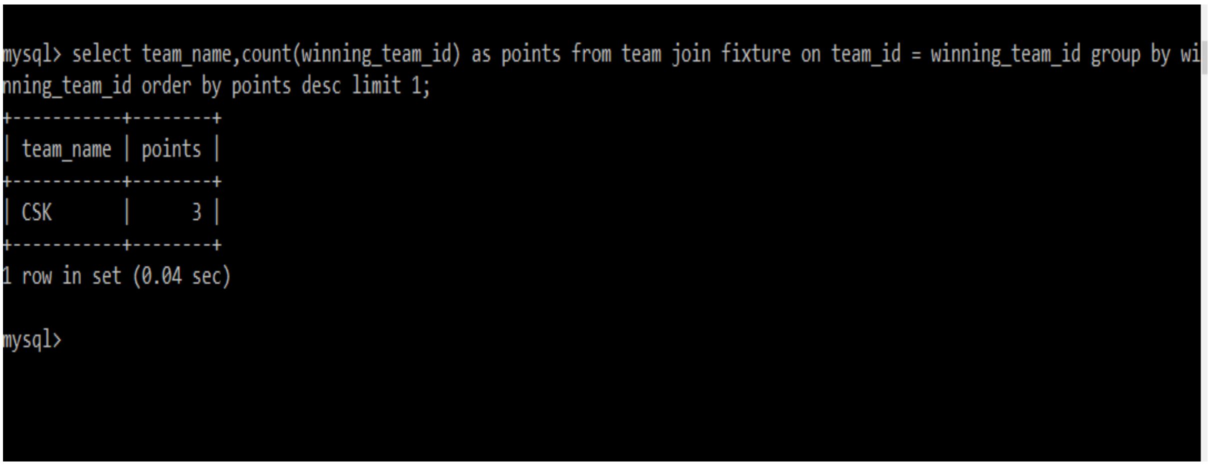
1)Vaasu

How many matches each team has won?



```
mysql> update team set team_name='RR' where team_name ='GG';
Query OK, 1 row affected (0.22 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select team_name,count(winning_team_id) as points from team join fixture on team_id = winning_team_id group by wi
nning_team_id;
+-----------+--------+
| team_name | points |
+-----------+--------+
| RR        |      2 |
| MI        |      1 |
| CSK       |      3 |
+-----------+--------+
3 rows in set (0.00 sec)

mysql>
```

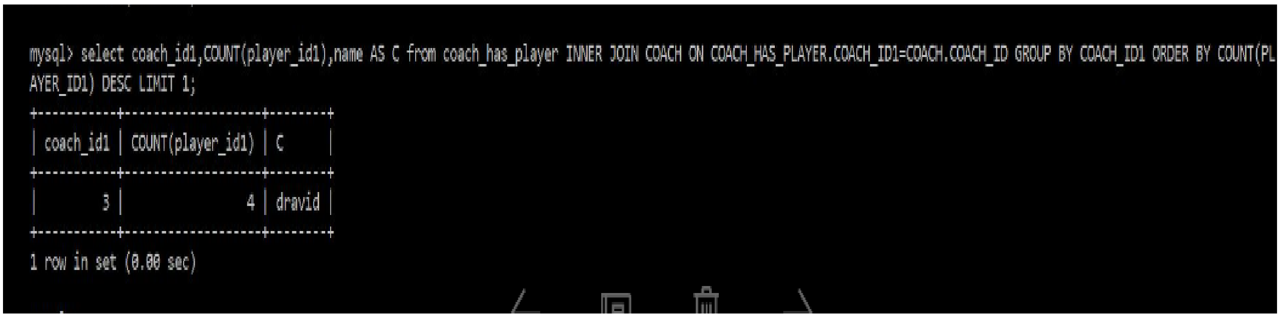
Which team has won maximum n.o of games?



```
mysql> select team_name,count(winning_team_id) as points from team join fixture on team_id = winning_team_id group by wi
nning_team_id order by points desc limit 1;
+-----------+--------+
| team_name | points |
+-----------+--------+
| CSK       |      3 |
+-----------+--------+
1 row in set (0.04 sec)

mysql>
```

2)Aryan

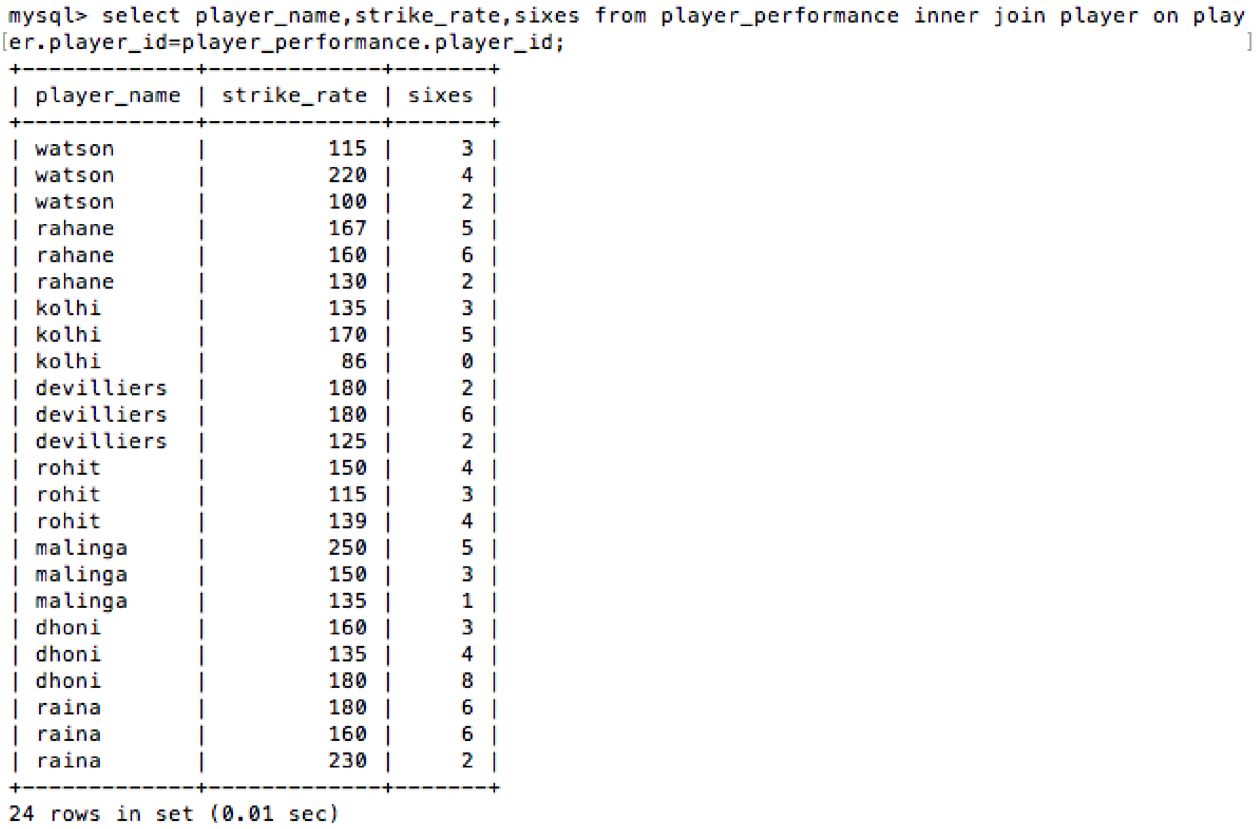
Which coach has coached maximum n.o of players give his name?



```
mysql> select coach_id1,COUNT(player_id1),name AS C from coach_has_player INNER JOIN COACH ON COACH_HAS_PLAYER.COACH_ID1=COACH.COACH_ID GROUP BY COACH_ID1 ORDER BY COUNT(PL
AYER_ID1) DESC LIMIT 1;
+-----------+-------------------+--------+
| coach_id1 | COUNT(player_id1) | C      |
+-----------+-------------------+--------+
|         3 |                 4 | dravid |
+-----------+-------------------+--------+
1 row in set (0.00 sec)
```

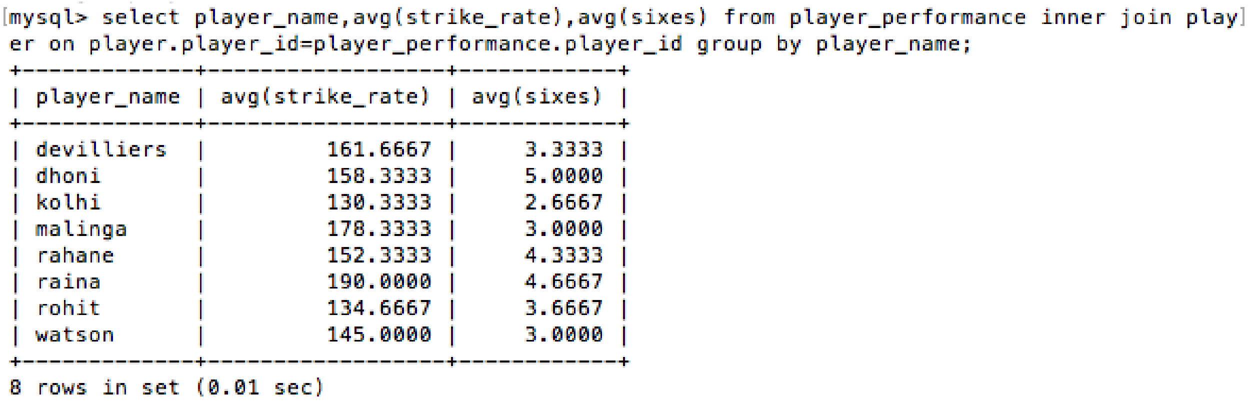
3)Akshat

Player wise strike rate and sixes:



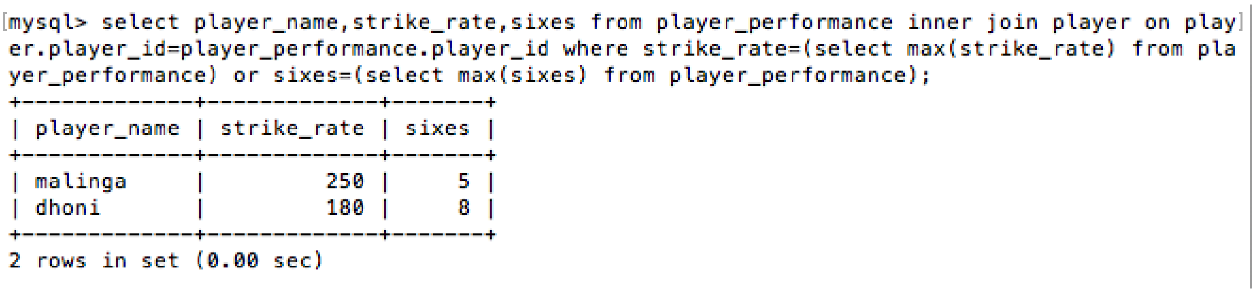
```
mysql> select player_name,strike_rate,sixes from player_performance inner join player on play
er.player_id=player_performance.player_id;
+-------------+-------------+-------+
| player_name | strike_rate | sixes |
+-------------+-------------+-------+
| watson      |         115 |     3 |
| watson      |         220 |     4 |
| watson      |         100 |     2 |
| rahane      |         167 |     5 |
| rahane      |         160 |     6 |
| rahane      |         130 |     2 |
| kolhi       |         135 |     3 |
| kolhi       |         170 |     5 |
| kolhi       |          86 |     0 |
| devilliers  |         180 |     2 |
| devilliers  |         180 |     6 |
| devilliers  |         125 |     2 |
| rohit       |         150 |     4 |
| rohit       |         115 |     3 |
| rohit       |         139 |     4 |
| malinga     |         250 |     5 |
| malinga     |         150 |     3 |
| malinga     |         135 |     1 |
| dhoni       |         160 |     3 |
| dhoni       |         135 |     4 |
| dhoni       |         180 |     8 |
| raina       |         180 |     6 |
| raina       |         160 |     6 |
| raina       |         230 |     2 |
+-------------+-------------+-------+
24 rows in set (0.01 sec)
```

Average strike rate and sixes:



```
mysql> select player_name,avg(strike_rate),avg(sixes) from player_performance inner join play
er on player.player_id=player_performance.player_id group by player_name;
+-------------+------------------+------------+
| player_name | avg(strike_rate) | avg(sixes) |
+-------------+------------------+------------+
| devilliers  |         161.6667 |     3.3333 |
| dhoni       |         158.3333 |     5.0000 |
| kolhi       |         130.3333 |     2.6667 |
| malinga     |         178.3333 |     3.0000 |
| rahane      |         152.3333 |     4.3333 |
| raina       |         190.0000 |     4.6667 |
| rohit       |         134.6667 |     3.6667 |
| watson      |         145.0000 |     3.0000 |
+-------------+------------------+------------+
8 rows in set (0.01 sec)
```

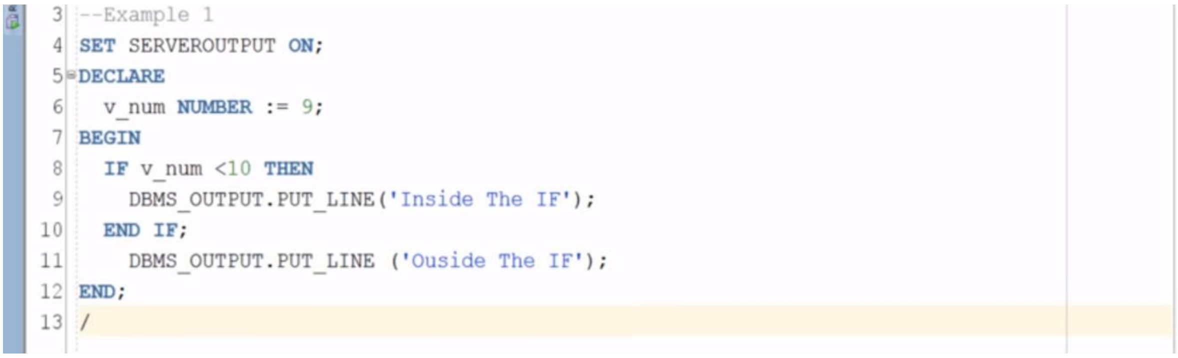
player with max strike rate and max sixes:



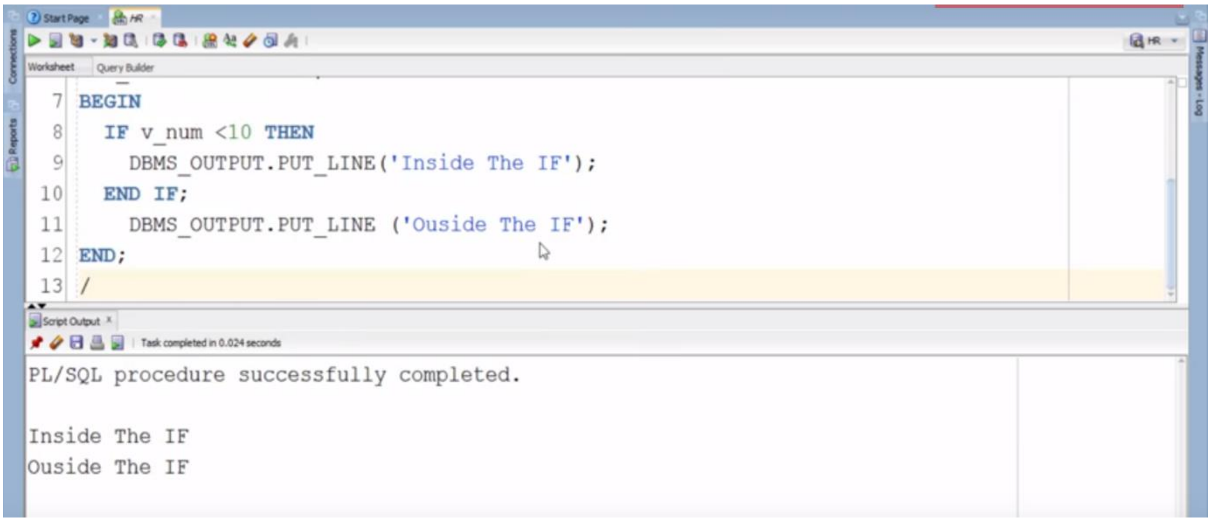
```
mysql> select player_name,strike_rate,sixes from player_performance inner join player on play
er.player_id=player_performance.player_id where strike_rate=(select max(strike_rate) from pla
yer_performance) or sixes=(select max(sixes) from player_performance);
+-------------+-------------+-------+
| player_name | strike_rate | sixes |
+-------------+-------------+-------+
| malinga     |         250 |     5 |
| dhoni       |         180 |     8 |
+-------------+-------------+-------+
2 rows in set (0.00 sec)
```

PL SQL

IF THEN



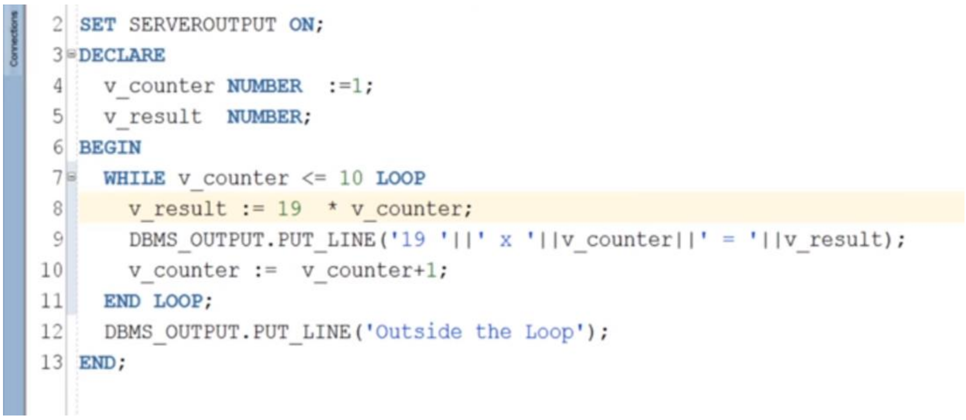
```
--Example 1
SET SERVEROUTPUT ON;
DECLARE
  v_num NUMBER := 9;
BEGIN
  IF v_num <10 THEN
    DBMS_OUTPUT.PUT_LINE('Inside The IF');
  END IF;
    DBMS_OUTPUT.PUT_LINE ('Ouside The IF');
END;
/
```



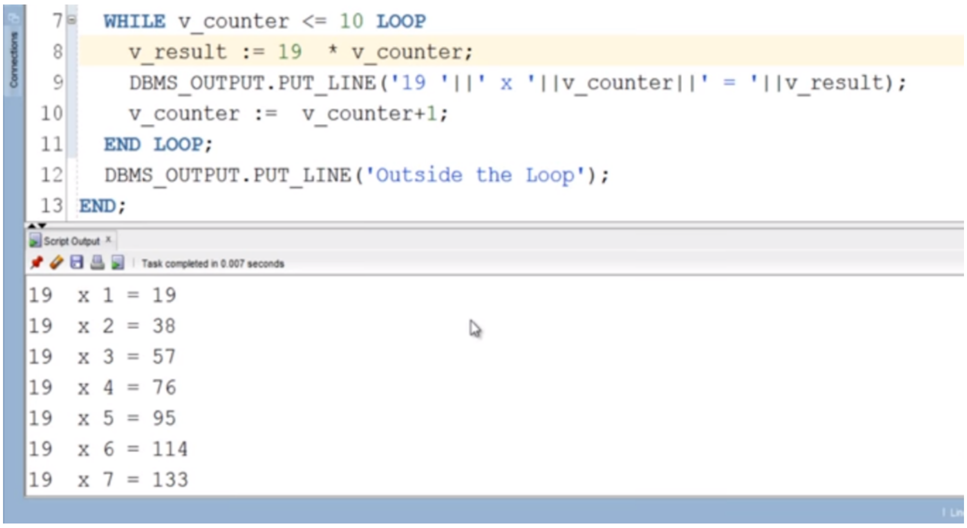
```
BEGIN
  IF v_num <10 THEN
    DBMS_OUTPUT.PUT_LINE('Inside The IF');
  END IF;
    DBMS_OUTPUT.PUT_LINE ('Ouside The IF');
END;
/
```

```
PL/SQL procedure successfully completed.

Inside The IF
Ouside The IF
```

WHILE LOOP



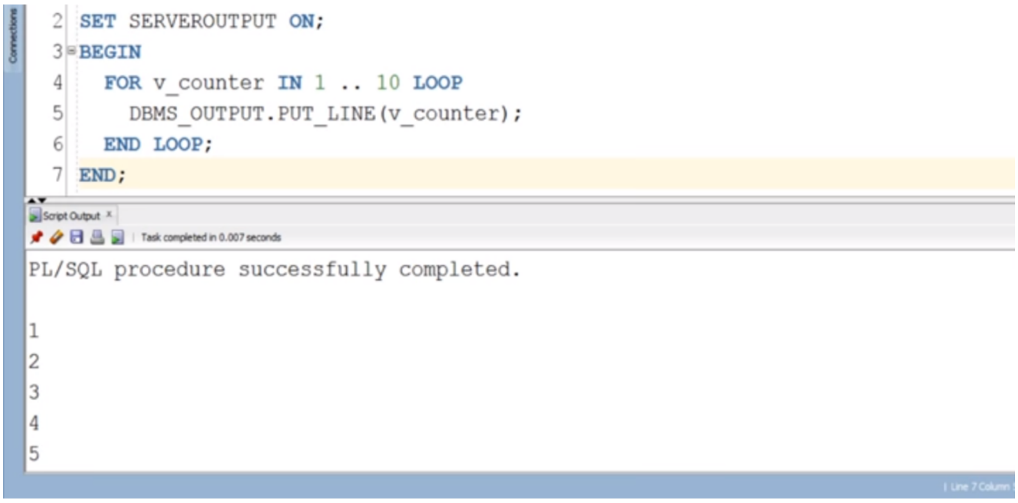
```
SET SERVEROUTPUT ON;
DECLARE
  v_counter NUMBER  :=1;
  v_result  NUMBER;
BEGIN
  WHILE v_counter <= 10 LOOP
    v_result := 19  * v_counter;
    DBMS_OUTPUT.PUT_LINE('19 '||' x '||v_counter||' = '||v_result);
    v_counter :=  v_counter+1;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('Outside the Loop');
END;
```



```
  WHILE v_counter <= 10 LOOP
    v_result := 19  * v_counter;
    DBMS_OUTPUT.PUT_LINE('19 '||' x '||v_counter||' = '||v_result);
    v_counter :=  v_counter+1;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('Outside the Loop');
END;
```

```
19  x 1 = 19
19  x 2 = 38
19  x 3 = 57
19  x 4 = 76
19  x 5 = 95
19  x 6 = 114
19  x 7 = 133
```

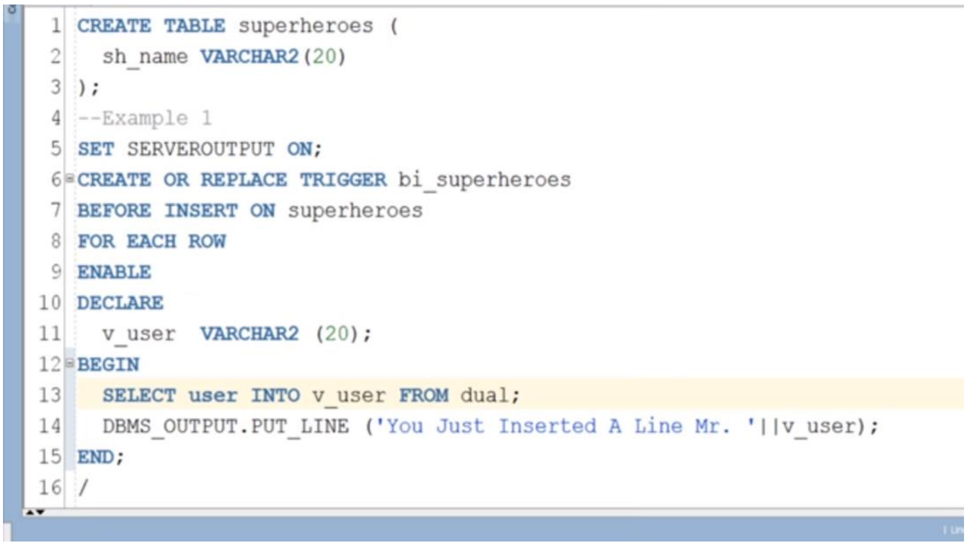
FOR LOOP:



```
SET SERVEROUTPUT ON;
BEGIN
  FOR v_counter IN 1 .. 10 LOOP
    DBMS_OUTPUT.PUT_LINE(v_counter);
  END LOOP;
END;
```

```
PL/SQL procedure successfully completed.

1
2
3
4
5
```

TRIGGER:



```
CREATE TABLE superheroes (
  sh_name VARCHAR2(20)
);
--Example 1
SET SERVEROUTPUT ON;
CREATE OR REPLACE TRIGGER bi_superheroes
BEFORE INSERT ON superheroes
FOR EACH ROW
ENABLE
DECLARE
  v_user  VARCHAR2 (20);
BEGIN
  SELECT user INTO v_user FROM dual;
  DBMS_OUTPUT.PUT_LINE ('You Just Inserted A Line Mr. '||v_user);
END;
/
```

Normalization of the database

If player details are included in player performance and the team details are included in team performance, then the table will only be normalized to 2NF.

Therefore the player performance and team performance details are filled in the constraint table formed by the m:n relation between fixture and performance tables.

Now the table is 3NF normalized.

Conclusion

We have created a database with our business rules in mind and after at least 10 tries we managed to normalise it to 3ed Normal Form. This database has till this point satisfied every query asked of it. Making this database has been really fun especially figuring out the possibilities that MySQL Workbench offers us.

We encountered many errors in the way and debugging them has shown how strict the constraints in DBMS have to be to run a successful query. After completing this project, we all have confidence in our database abilities and look forward to exploring this field.