\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CREATE TABLE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

CREATE TABLE NATİON (

NATİON\_ID varchar(50) PRIMARY KEY ,

NATİON\_NAME varchar(50)

);

CREATE TABLE TEAM (

TEAM\_ID int PRIMARY KEY,

TEAM\_NAME varchar(50) ,

TEAM\_group varchar(5)

);

CREATE TABLE GOAL (

GOAL\_ID int PRIMARY KEY,

F\_PLAYER\_NAME varchar(50) ,

FF\_PLAYER\_ID int ,

M\_MATCH\_ID int ,

FOREIGN KEY (FF\_PLAYER\_ID) REFERENCES football\_PLAYER (F\_PLAYER\_ID),

FOREIGN KEY (M\_MATCH\_ID) REFERENCES MATCH (MATCH\_ID)

);

-- Table: MANAGER

CREATE TABLE MANAGER (

MANAGER\_ID int PRIMARY KEY,

MANAGER\_NAME varchar(50) ,

MANAGER\_COST int ,

N\_NATİON\_ID varchar(50) ,

FOREIGN KEY (N\_NATİON\_ID) REFERENCES NATİON (NATİON\_ID)

);

CREATE TABLE employees (

Empid int PRIMARY KEY,

EName varchar(50) ,

ESurname varchar(50) ,

ESalary int ,

EComm int,

MM\_MANAGER\_ID int,

FOREIGN KEY (MM\_MANAGER\_ID) REFERENCES MANAGER (MANAGER\_ID)

);

CREATE TABLE MATCH (

MATCH\_ID int PRIMARY KEY,

MATCH\_DATE date ,

hTEAM\_ID int ,--home team

aTEAM\_ID int ,--away team

SCORE int ,

FOREIGN KEY (hTEAM\_ID) REFERENCES TEAM (TEAM\_ID)

);

CREATE TABLE TEAM\_MANAGER (

TEAM\_MANAGER\_ID int PRIMARY KEY,

START\_DATE date ,

END\_DATE date ,

MMANAGER\_ID int ,

TTEAM\_ID int,

FOREIGN KEY (MMANAGER\_ID) REFERENCES MANAGER (MANAGER\_ID),

FOREIGN KEY (TTEAM\_ID) REFERENCES TEAM (TEAM\_ID)

);

CREATE TABLE football\_PLAYER (

F\_PLAYER\_ID int PRIMARY KEY,

F\_PLAYER\_NAME varchar(50) ,

TTEAM\_ID int,

N\_NATİON\_ID varchar(50),

FOREIGN KEY (TTEAM\_ID) REFERENCES TEAM (TEAM\_ID),

FOREIGN KEY (N\_NATİON\_ID) REFERENCES NATİON (NATİON\_ID)

);

CREATE TABLE sponsor (

Sponsor\_ID int PRIMARY KEY,

Sponsor\_name varchar(50),

Starting\_of\_sponsor date,

price int ,

TTEAM\_ID int ,

FOREIGN KEY (TTEAM\_ID) REFERENCES TEAM (TEAM\_ID)

);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ORACLE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

----------------------------------------------------procedure--------------------------------------------------------------------------------

--1

drop procedure SponsorS;

CREATE OR REPLACE PROCEDURE SponsorS

AS

NOPlayStationNOSAMSUNG EXCEPTION;

X NUMBER;

IDS NUMBER;

CURSOR spoid IS

SELECT Sponsor\_ID

FROM sponsor

WHERE Sponsor\_name IN('PlayStation','SAMSUNG');

BEGIN

SELECT COUNT(Sponsor\_ID) INTO X

FROM sponsor

WHERE Sponsor\_name IN('PlayStation','SAMSUNG');

IF X = 0 THEN RAISE NOPlayStationNOSAMSUNG;

ELSE

OPEN spoid;

LOOP

FETCH spoid INTO IDS;

EXIT WHEN spoid%NOTFOUND;

UPDATE sponsor SET price = price\*1.1

WHERE Sponsor\_ID = IDS AND Sponsor\_name = 'PlayStation';

UPDATE sponsor SET price = price\*1.2

WHERE Sponsor\_ID = IDS AND Sponsor\_name = 'SAMSUNG';

END LOOP;

CLOSE spoid;

END IF;

EXCEPTION WHEN NOPlayStationNOSAMSUNG THEN

DBMS\_OUTPUT.PUT\_LINE('NOBODY sponsor AS PlayStation or SAMSUNG');

END;

--2 CURSOR EKLE

CREATE TABLE MatchaRequests

(MatchaRequestsid NUMBER(4),

teamneeds VARCHAR(13),

matchDATE DATE,

teamWHO NUMBER(4));

Drop procedure AddMatch

CREATE OR REPLACE PROCEDURE AddMatch(teamneeds VARCHAR, matchDATE DATE, teamWHO NUMBER)

AS

noMatch EXCEPTION;

numberOfMatche NUMBER;

idNewMatch NUMBER;

BEGIN

SELECT COUNT(MATCH\_ID) INTO numberOfMatche

FROM MATCH

WHERE MATCH\_ID = teamWHO;

IF numberOfMatche = 0 THEN RAISE noMatch;

ELSE

SELECT NVL(MAX(MatchaRequestsid) + 1, 1) INTO idNewMatch

FROM MatchaRequests;

INSERT INTO MatchaRequests VALUES

(idNewMatch, teamneeds, matchDATE, teamWHO);

END IF;

EXCEPTION WHEN noMatch THEN

dbms\_output.put\_line( teamWHO );

END;

--3 Add bonus based on value given to football player price with name

Drop procedure bonusPrice

CREATE OR REPLACE PROCEDURE bonusPrice(cName varchar, bonus int)

As

Nofootballp Exception;

CURSOR person IS

Select F\_PLAYER\_ID

From football\_PLAYER

Where F\_PLAYER\_NAME = cName;

counting Number;

idF Number;

Begin

Select COUNT(F\_PLAYER\_ID) into counting

From football\_PLAYER

Where F\_PLAYER\_NAME = cName;

IF counting = 0 Then raise Nofootballp;

Else

Open person;

Loop

Fetch person into idF;

Exit When person%notfound;

Update football\_PLAYER

Set price = price+bonus

Where F\_PLAYER\_ID = idF;

End Loop;

dbms\_output.put\_line('price of futbool player ' || cName || ' raised by' || bonus);

END IF;

Exception When Nofootballp Then

dbms\_output.put\_line('There are no such player');

END;

select F\_PLAYER\_NAME, price

from football\_PLAYER

where F\_PLAYER\_NAME = 'Ugurcan Cakir';

Exec bonusPrice('Ugurcan Cakir',200);

--4 Add new booking for the guest with given id to the given room which its clerk will be the one with highest salary.

CREATE OR REPLACE PROCEDURE AddGoal(idmhatch int, idR int)

As

NoPlayer Exception;

CURSOR football\_PLAYER IS

Select F\_PLAYER\_ID

From football\_PLAYER

Where price in (Select MAX(price) from football\_PLAYER);

countM Number;

countR Number;

idNewGoal Number;

idP Number;

Begin

Select COUNT(MATCH\_ID) into countM

from MATCH

Where MATCH\_ID = idmhatch;

IF countM = 0 Then raise NoPlayer;

ELSE

Open football\_PLAYER;

Loop

Fetch football\_PLAYER into idP;

Exit When football\_PLAYER%notfound;

Select NVL(MAX(GOAL\_ID)+1,1) into idNewGoal

From GOAL;

INSERT INTO GOAL VALUES(idNewGoal,idmhatch,idR,idP);

End Loop;

END IF;

Exception When NoPlayer Then dbms\_output.put\_line('There is no such player');

END;

---------------------------------------------------------TRIGGER---------------------------------------------------------------------------------------------

--1

DROP TRIGGER Manager\_Of\_Cost

SET SERVEROUTPUT ON

CREATE OR REPLACE TRIGGER Manager\_Of\_Cost

BEFORE UPDATE ON MANAGER

FOR EACH ROW

DECLARE

v\_MANAGER\_NAME Varchar;

v\_MANAGER\_ID int;

M\_COST Number(8,2) := 0;

CURSOR Cur\_MANAGER IS SELECT MANAGER\_ID, MANAGER\_NAME FROM MANAGER;

BEGIN

OPEN Cur\_MANAGER;

LOOP

FETCH Cur\_MANAGER INTO v\_MANAGER\_ID, v\_MANAGER\_NAME;

EXIT WHEN v\_MANAGER\_NAME%NOTFOUND;

SELECT SUM (NVL(MANAGER\_COST),0) INTO M\_COST

FROM MANAGER WHERE MANAGER\_ID = v\_MANAGER\_ID;

dbms\_output.put\_line( M\_COST );

M\_COST := 0;

END LOOP;

CLOSE Cur\_MANAGER;

END;

--2 Trigger currently reserved when team manager changes

DROP TRIGGER changes\_managemnt\_team\_trig

create or replace trigger changes\_managemnt\_team\_trig

before delete

on TEAM

for each row

declare

cursor a1 is

select \* from TEAM\_MANAGER;

begin

for TEAM\_MANAGER in a1 loop

if TEAM\_MANAGER.TTEAM\_ID = :OLD.TEAM\_ID and TEAM\_MANAGER.START\_DATE < sysdate

then RAISE\_APPLICATION\_ERROR(-20000,'Cant uninstall, maneger has team now.');

end if;

end loop;

end;

select \*

from TEAM\_MANAGER;

delete from TEAM

where TEAM\_ID = 1201;

--3 Trigger which is enabled when creating a booking on a room that is already booked at the moment.

DROP TRIGGER sponsor\_for\_same\_team\_trig

create or replace trigger sponsor\_for\_same\_team\_trig

before insert

on sponsor

for each row

declare

x number;

currentDate date;

cursor a1 is

select \* from TEAM;

begin

currentDate := :new.Starting\_of\_sponsor;

select count(Sponsor\_ID) into x

from sponsor

where Starting\_of\_sponsor = currentDate;

for TM in a1 loop

if :new.TTEAM\_ID = TM.TEAM\_ID and x > 0

then RAISE\_APPLICATION\_ERROR(-20000, 'Team Number ' || TM.TEAM\_ID || ' has already been scheduled for the same date.');

end if;

end loop;

end;

Select F\_PLAYER\_ID

from football\_PLAYER

where hTEAM\_ID

MATCH\_ID

--4 trigger that increases up to 10% to players playing in the match.

create or replace trigger raise\_match\_pri\_trigger

after insert on MATCH

for each row

declare

cursor pl is

select F\_PLAYER\_ID from football\_PLAYER;

currentID int;

idF int;

idT int;

begin

open pl;

idT := :new.hTEAM\_ID;

Select F\_PLAYER\_ID into idF

from football\_PLAYER

where hTEAM\_ID = idT;

loop

fetch pl into currentID;

exit when pl%notfound;

if currentID = idF

then

Update football\_PLAYER

Set price = price\*1.1

Where F\_PLAYER\_ID = idF;

DBMS\_OUTPUT.PUT\_LINE('Price of football Player with id ' || idF || ' raised by 10%.');

end if;

end loop;

close pl;

end;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*tsql\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--TRIGGER

--1

create trigger HighmanEmp

on Employees for update as

Declare @D\_bud INT;

Declare @D\_id INT;

Select @D\_bud = SUM(ESalary+100) From employees, MANAGER

where employees.Empid = employees.MM\_MANAGER\_ID and MANAGER.MANAGER\_ID=01;

Select @D\_id = empid From employees, MANAGER

where employees.Empid = employees.MM\_MANAGER\_ID and MANAGER.MANAGER\_ID=01;

begin

if @D\_bud <= 5000

Update employees

Set ESalary = ESalary+100

Where empid = @D\_id;

print 'Higher salaries in Management';

end;

--2--2 Trigger currently reserved when team manager changes

DROP TRIGGER changes\_managemnt\_team\_trig

GO

CREATE TRIGGER changes\_managemnt\_team\_trig ON TEAM FOR BEFORE DELETE

DECLARE A1 CURSOR FOR SELECT \*FROM TEAM\_MANAGER;

BEGIN

DECLARE

OPEN A1

FETCH NEXT FROM A1 INTO

create or replace trigger changes\_managemnt\_team\_trig

before delete

on TEAM

for each row

declare

cursor a1 is

select \* from TEAM\_MANAGER;

begin

for TEAM\_MANAGER in a1 loop

if TEAM\_MANAGER.TTEAM\_ID = :OLD.TEAM\_ID and TEAM\_MANAGER.START\_DATE < sysdate

then RAISE\_APPLICATION\_ERROR(-20000,'Cant uninstall, maneger has team now.');

end if;

end loop;

end;

select \*

from TEAM\_MANAGER;

delete from TEAM

where TEAM\_ID = 1201;

---------------------------------------------procedure--------------------------------

--1

Create procedure SumofMatch (@futPlayername varchar)

as

begin

declare @sumOfMatch int;

Set nocount on;

select @sumOfMatch = sum(M\_MATCH\_ID)

from GOAL G,football\_PLAYER F

where f.F\_PLAYER\_ID=g.FF\_PLAYER\_ID ;

print @sumOfMatch;

End;

--2

CREATE PROCEDURE SponsorS

AS

BEGIN

DECLARE @IDS int;

DECLARE spoid CURSOR LOCAL FOR

SELECT Sponsor\_ID

FROM sponsor

WHERE Sponsor\_name IN('PlayStation','SAMSUNG');

SET NOCOUNT ON;

OPEN spoid;

WHILE 1=1 BEGIN

FETCH spoid INTO @IDS;

IF @@FETCH\_STATUS <> 0 BREAK ;

UPDATE sponsor SET price = price\*1.1

WHERE Sponsor\_ID = @IDS AND Sponsor\_name = 'PlayStation';

UPDATE sponsor SET price = price\*1.2

WHERE Sponsor\_ID = @IDS AND Sponsor\_name = 'SAMSUNG';

END;

CLOSE spoid;

DEALLOCATE spoid;

END;

--3

CREATE PROCEDURE bonusPrice(@cName varchar, @bonus int)

As

Begin

DECLARE person CURSOR LOCAL FOR

Select F\_PLAYER\_ID

From football\_PLAYER

Where F\_PLAYER\_NAME = @cName;

Declare @counting int;

Declare @idF int;

Set nocount on;

Select @counting = COUNT(F\_PLAYER\_ID)

From football\_PLAYER

Where F\_PLAYER\_NAME = @cName;

IF @counting = 0 Begin raise Nofootballp;

End

Else Begin

Open person;

While 1=1 begin

Fetch person into @idF;

If @@FETCH\_STATUS <> 0 BREAK ;

Update football\_PLAYER

Set price = price+@bonus

Where F\_PLAYER\_ID = @idF;

End;

print 'price of futbool player ' + ISNULL(@cName, '') + ' raised by' + isnull(@bonus, '');

END

Exception When Nofootballp Then

print 'There are no such player';

END;

--4

GO

CREATE PROCEDURE AddGoal(@idmhatch int, @idR int)

As

Begin

DECLARE football\_PLAYER CURSOR LOCAL FOR

Select F\_PLAYER\_ID

From football\_PLAYER

Where price in (Select MAX(price) from football\_PLAYER);

Declare @countM int;

Declare @countR int;

Declare @idNewGoal int;

Declare @idP int;

Set nocount on;

Select @countM = COUNT(MATCH\_ID)

from MATCH

Where MATCH\_ID = @idmhatch;

IF @countM = 0 Begin raise NoPlayer;

END

ELSE BEGIN

Open football\_PLAYER;

While 1=1 begin

Fetch football\_PLAYER into @idP;

If @@FETCH\_STATUS <> 0 BREAK ;

Select @idNewGoal = ISNULL(MAX(GOAL\_ID)+1,1)

From GOAL;

INSERT INTO GOAL VALUES(@idNewGoal,@idmhatch,@idR,@idP);

End;

END

Exception When NoPlayer Then print 'There is no such player';

END;