Антоанета Рафаилова Иванова, F77618, I семестър

Софтуерни технологии в Интернет Магистърска програма,

INFM111, INFM160

Тема:

Спортни състезания и участници

Базата данни е създадена с цел да предостави информация за спортни състезания, участниците в тях и за това кога те се провеждат и на какво място.

Базата съдържа три таблици, първата таблица има име Activities, носи информация относно провеждането на събитията. Съдържа следните атрибути: ActivitiesID, която колона носи информация от тип Number, представлява първичен ключ за таблицата и на всяка категория спорт съответства уникален номер от тази колона. Следващия атрибут е Categories, където са описани различните категории събития, които ще бъдат спонсорирани, данните , които съдържа са от тип Text. Таблицата съдържа две колони EndDate и StartDate, които носят информация от тип Date за началната и крайната дата на всяко събитие. Колоната Type съдържа информация за това какъв е типа на провежданото събитие, а именно дали то е на закрито или открито и данните в нея са от тип Text. Колоната Location съдържа данни от тип Text и носи информация за мястото където ще се проведе събитието.

```
CREATE TABLE  "ACTIVITIES"

  (        "ACTIVITIES_ID" NUMBER(38,0) GENERATED ALWAYS AS IDENTITY MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER  NOCYCLE
NOT NULL ENABLE,

        "CATHEGORIES" VARCHAR2(225),

        "STARTDATE" DATE,

        "ENDDATE" DATE,

        "TYPE" VARCHAR2(225),

        "LOCATION" VARCHAR2(225),

         CONSTRAINT "ACTIVITIES_PK" PRIMARY KEY ("ACTIVITIES_ID")

 USING INDEX  ENABLE

  )

/
```

Втората таблица е съдържа данни за участниците във всяко събитие**.** Първата и колона е ID, която е първичен ключ за тази таблица, представлява уникален номер и данните са от тип Number, следващата колона е Team, данните в нея са от тип Number, и носи информация за броя на участниците във всеки отбор, следващата колона е Age, данните в нея са от тип Number и носи инфомация за възрастта на участниците във всеки отбор, която е допустима за участие в състезанията. Следващата колона е Country, данните в нея са от тип Text и носи информация за това от коя страна е всеки отбор.

```
CREATE TABLE  "PARTICIPANTS"
```

```
  (        "ID" NUMBER(38,0) GENERATED ALWAYS AS IDENTITY MINVALUE 1 MAXVALUE
99999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER  NOCYCLE
NOT NULL ENABLE,
        "TEAM" NUMBER(38,0) NOT NULL ENABLE,
        "AGE" NUMBER(38,0) NOT NULL ENABLE,
        "COUNTRY" VARCHAR2(225) NOT NULL ENABLE,
         CONSTRAINT "PARTICIPANTS_PK" PRIMARY KEY ("ID")
 USING INDEX  ENABLE
  )
/
```

Третата таблица се нарича ACTIVITIES_DETAILS съдържа три колони- ACTN_UM, NEWLOCATION,BRANDS, колоната ACT_NUM данните в тях са от тип NUMBER и VARCHAR . Таблиците не са свързани по между си.

```
CREATE TABLE  "ACTIVITIES_DETAILS"
  (        "ACT_NUM" NUMBER(38,0) NOT NULL ENABLE,
        "NEWLOCATION" VARCHAR2(225),
        "BRANDS" VARCHAR2(225),
         CONSTRAINT "ACTIVITIES_DETAILS_PK" PRIMARY KEY ("ACT_NUM")
 USING INDEX  ENABLE
  )
/
```

1. Тази функция е без параметри, връща променлива от тип номер и това, което прави е да показва коя е максималната възраст на участник в състезание.

Таблицата върху, която се изпълнява е PARTICIPANTS, колоната е AGE.

```
CREATE OR REPLACE FUNCTION maxAge
RETURN NUMBER
IS
num PARTICIPANTS.AGE%TYPE;
BEGIN
SELECT MAX(AGE) INTO num
```

FROM PARTICIPANTS;

RETURN num;

END;

Извиква се с:

BEGIN

DBMS_OUTPUT.PUT_LINE(maxAge);

END;

Резултат :

```
32

Statement processed.
```

2. Следващата процедура, показва началната дата на състезанията по зададено ID, Изпълнява се върху таблица ACTIVITIES, и приема един входен параметър тип номер.

create or replace procedure empshow2(eno number)

is

act_date ACTIVITIES.STARTDATE%TYPE;

BEGIN

select STARTDATE INTO act_date from ACTIVITIES where ACTIVITIES_ID = eno;

DBMS_OUTPUT.PUT_LINE(act_date);

END;

Извиква се с:

begin

empshow2(62);

end;

резултат

```
12/05/2017

Statement processed.
```

0.05 seconds

3.Следващата процедура, показва всеки ред от таблицата, който съответства на зададен номер на ID, Изпълнява се върху таблица ACTIVITIES, и приема един входен параметър тип номер.

create or replace procedure empshow3(eno number)

is

act_date ACTIVITIES%ROWTYPE;

BEGIN

select * INTO act_date from ACTIVITIES where ACTIVITIES_ID = eno;DBMS_OUTPUT.PUT_LINE(act_date.CATHEGORIES||' '|| act_date.STARTDATE||' '|| act_date.ENDDATE||' '|| act_date.LOCATION||' '||act_date.TYPE);

END;

Извиква се с

begin

empshow3(62);

end;

резултат:

```
Skateboarding      12/05/2017     12/06/2017     Greece     Indoors

Statement processed.
```

0.02 seconds

4. Следващата процедура съдържа курсор и отпечатва при извикването и данните съхранени в курсора от колоната TEAM. Няма параметри.

CREATE OR REPLACE PROCEDURE showpart

IS

CURSOR bla IS SELECT * FROM PARTICIPANTS;

part_rec PARTICIPANTS%ROWTYPE;

BEGIN

OPEN bla;

LOOP

FETCH bla INTO part_rec;

EXIT WHEN bla%NOTFOUND;

DBMS_OUTPUT.PUT_LINE(part_rec.TEAM);

END LOOP;

CLOSE bla;

END;

BEGIN (извикването се изпълнява тук)

showpart;

END;

5. Процедурата showAge, приема един параметър, който е номер от ID колоната, и има една променлива , процедурата съдържа условие,на подаденият като аргумент номер да съответства стойност от колоната AGE , по-малка от 32, ако това е така, тя ще бъде отпечатана.

```
CREATE OR REPLACE PROCEDURE showAge(eno NUMBER)
IS
age_no PARTICIPANTS.AGE%TYPE;
BEGIN

SELECT age INTO age_no
FROM  PARTICIPANTS
WHERE ID=eno;
IF age_no < 32
THEN
DBMS_OUTPUT.PUT_LINE( age_no );
END IF;
END;
```

Извиква се с :

```
BEGIN
showAge(1);
END;
```

6. Triggers Тригерът , който съм създала се изпълнява за таблица ACTIVITIES_DETAILS, като добавя нов ред, в таблицата със стойностите, които са подадени.

```
CREATE OR REPLACE TRIGGER act_2
BEFORE
INSERT
ON ACTIVITIES_DETAILS
FOR EACH ROW
BEGIN
DBMS_OUTPUT.PUT_LINE('Before inserting ' || :new.BRANDS);
```

```
END;
INSERT INTO ACTIVITIES_DETAILS
VALUES('4','AMSTERDAM', 'vllm');
```

7. Създавам пакет с една функция. Първоначално създавам пакета и му давам име, обявявам какви функции ще има в него, само чрез техните имена и параметри , ако имат такива. След което съставям тялото на пакета, като в него е разписана функцията. Накрая декларирам променлива, в която съхранявам резултата от извикването на функцията, извикването на функцията става, чрез името на пакета и името на функцията.

```
 create package part_mng as
function maxAge return NUMBER;
end part_mng;

create package body part_mng as
FUNCTION maxAge
RETURN NUMBER
IS
num PARTICIPANTS.AGE%TYPE;
BEGIN
SELECT MAX(AGE) INTO num
FROM PARTICIPANTS;
RETURN num;
end maxAge;
end part_mng;

DECLARE
new_age NUMBER (38);
BEGIN
new_age := part_mng.maxAge;

DBMS_OUTPUT.PUT_LINE( new_age);
END;
```

8. Създавам пакет, в който има една функция и една процедура, като първата е тази в задача 7 , втората е процедура е процедурата showpart, която няма параметри. При създаването на пакета , тялото и викането на фунцкията и процедурата единственото различно от описанието в горния пример е, че освен функцията имаме и процедура, която трябва да присъства като име в създаването на пакета, тялото и извикването и.

```
create package part_mng as

function maxAge return NUMBER;

procedure showpart;

end part_mng;


create package body part_mng as

FUNCTION maxAge

RETURN NUMBER

IS

num PARTICIPANTS.AGE%TYPE;

BEGIN

SELECT MAX(AGE) INTO num

FROM PARTICIPANTS;

RETURN num;

end maxAge;


PROCEDURE showpart

IS

CURSOR bla IS SELECT * FROM PARTICIPANTS;

part_rec PARTICIPANTS%ROWTYPE;

BEGIN

OPEN bla;

LOOP

FETCH bla INTO part_rec;

EXIT WHEN bla%NOTFOUND;

DBMS_OUTPUT.PUT_LINE(part_rec.TEAM);

END LOOP;
```

```
CLOSE bla;
END showpart;
end part_mng;

DECLARE
new_age NUMBER (38);
BEGIN
part_mng.showpart;
new_age := part_mng.maxAge;
DBMS_OUTPUT.PUT_LINE( new_age);

END;
```