

BAZĂ DE DATE

**ORGANIZAȚIA
NONGUVERNAMENTALĂ
TOGETHER WE CAN**

Căpitănu Andreea
Grupa 243

1. Prezentați pe scurt baza de date (utilitatea ei).

"Together we can" este o asociație nonguvernamentală ce are ca scop realizarea de mici schimbări în viețile și mentalitățile oamenilor din comunitatea românească, prin proiecte cu rol informativ, educativ, social sau caritabil.

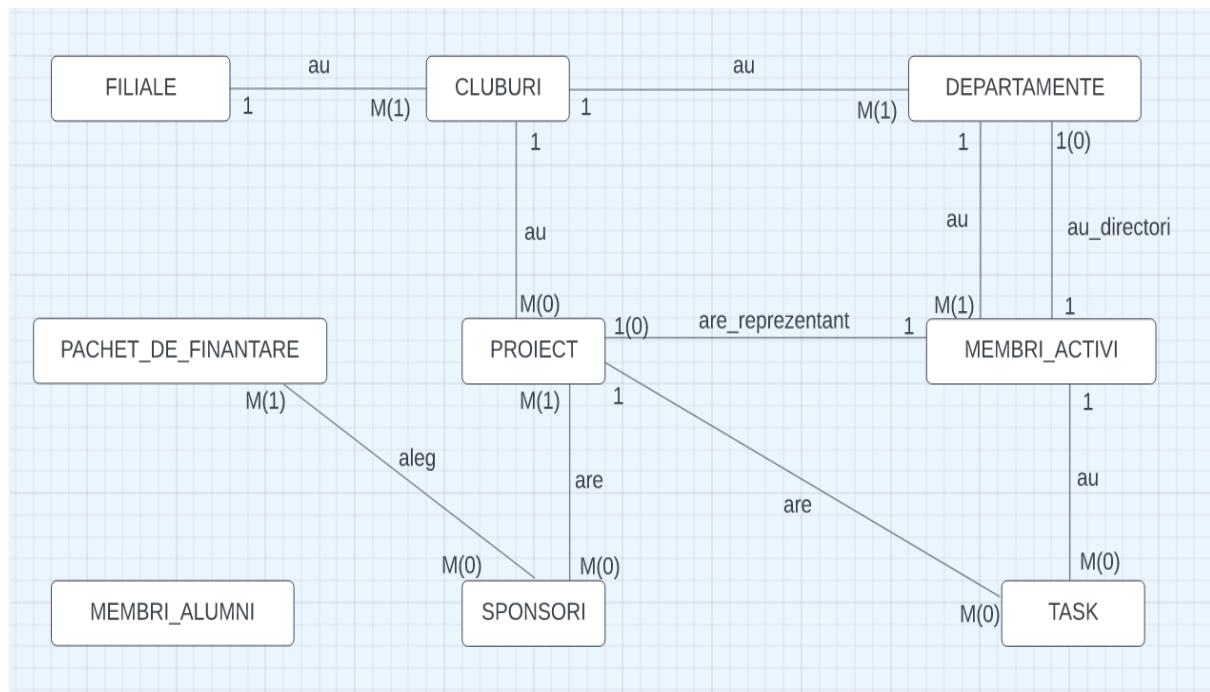
Asociația este formată din mai multe filiale, plasate în centrele de interes ale țării, care la rândul lor prezintă subfiliale, numite cluburi. Fiecare entitate a asociației este organizată pe departamente, din care fac parte membri activi. Pe lângă aceștia, Asociația "Together we can" acordă un respect remarcabil membrilor Alumni (foști membri ce au avut un impact incontestabil în dezvoltarea organizației), ale căror opinii și sfaturi sunt luate în considerare.

De-a lungul mandatelor, asociația își propune organizarea unor serii de proiecte. Pentru realizarea lor, este necesară formarea unei echipe de lucru, în structura căreia să fie un membru din fiecare departament. Totodată, în proiectele mai complexe, pentru o mai bună organizare a sarcinilor, taskurile sunt structurate pe sprinturi.

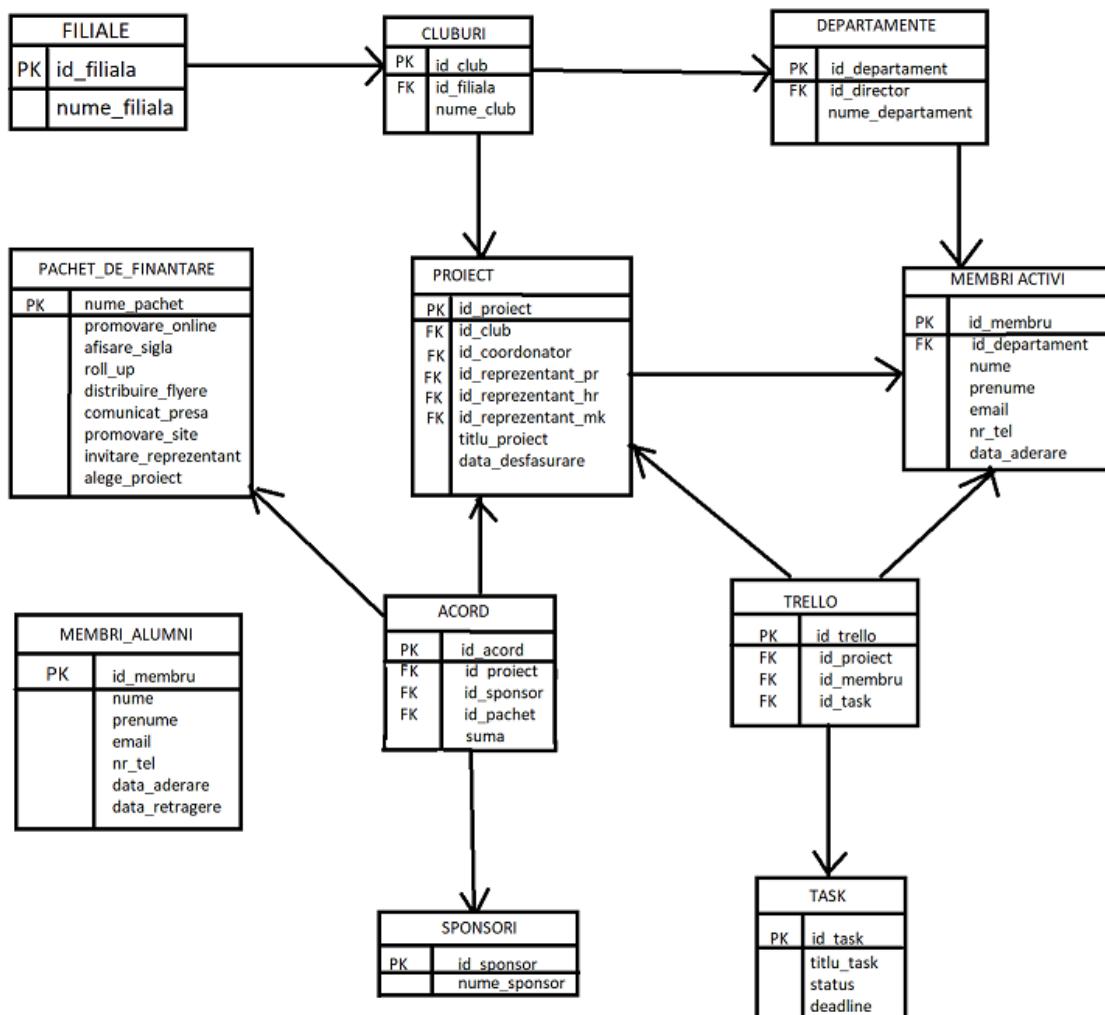
Luând în considerare faptul că asociația este una nonguvernamentală, pentru realizarea proiectelor este necesară contactarea potențialilor sponsori. Aceștia pot fi companii sau persoane fizice. În procesul contactării, le sunt prezentate pachetele de finanțare: sponsor, partener sau premium. Acestea suprind serviciile de care beneficiază sponsorii odată cu încheierea parteneriatului lor.

Astfel, prin intermediul bazei de date create, se urmărește gestionarea mai ușoară a membrilor organizației, eficientizarea muncii acestora, contorizarea sponsorilor și a pachetelor alese, cât și o mai bună organizare a proiectelor, prin reținerea materialelor necesare și a statusurilor taskurilor.

2. Realizați diagrama entitate-relație (ERD).



3. Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atrbutele necesare.



4. Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, definind toate constrângerile de integritate necesare (chei primare, cheile externe etc).

```

CREATE TABLE DEPARTAMENTE
  (ID_DEPARTAMENT NUMBER(4),
  NUME_DEPARTAMENT VARCHAR(30),
  ID_DIRECTOR NUMBER(4),
  ID_CLUB NUMBER(4),

  PRIMARY KEY(ID_DEPARTAMENT));
  
```

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_DEPARTAMENT	NUMBER(4, 0)	No	(null)	1	(null)
2 NUME_DEPARTAMENT	VARCHAR2(30 BYTE)	Yes	(null)	2	(null)
3 ID_DIRECTOR	NUMBER(4, 0)	Yes	(null)	3	(null)
4 ID_CLUB	NUMBER(4, 0)	Yes	(null)	4	(null)

```

CREATE TABLE MEMBRI_ACTIVI
(ID_MEMBRU NUMBER(4),
NUME VARCHAR(20) NOT NULL,
PRENUME VARCHAR(20),
EMAIL CHAR(30),
NR_TEL VARCHAR(15),
DATA_ADERARE DATE DEFAULT SYSDATE,
ID_DEPARTAMENT NUMBER(4),

PRIMARY KEY(ID_MEMBRU),
CONSTRAINT FK_ID_DEPARTAMENT FOREIGN KEY(ID_DEPARTAMENT)
REFERENCES DEPARTAMENTE(ID_DEPARTAMENT),
CONSTRAINT U_NUME_PRENUME UNIQUE(NUME, PRENUME),
CONSTRAINT U_EMAIL UNIQUE(EMAIL));

```

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_MEMBRU	NUMBER(4,0)	No	(null)	1	(null)
2 NUME	VARCHAR2(20 BYTE)	No	(null)	2	(null)
3 PRENUME	VARCHAR2(20 BYTE)	Yes	(null)	3	(null)
4 EMAIL	CHAR(30 BYTE)	Yes	(null)	4	(null)
5 NR_TEL	VARCHAR2(15 BYTE)	Yes	(null)	5	(null)
6 DATA_ADERARE	DATE	Yes	SYSDATE	6	(null)
7 ID_DEPARTAMENT	NUMBER(4,0)	Yes	(null)	7	(null)

```

CREATE TABLE PROIECT
(ID_PROIECT NUMBER(4),
TITLU_PROIECT VARCHAR(30),
SCOP VARCHAR(20),
DATA_DESFASURARE DATE,
REPREZENTANT_PROIECTE NUMBER(4),
REPREZENTANT_PR NUMBER(4),

```

```

REPREZENTANT_HR NUMBER(4),
REPREZENTANT_MK NUMBER(4),

PRIMARY KEY(ID_PROIECT),
CONSTRAINT FK_REPREZENTANT_PROIECTE FOREIGN KEY(REPREZENTANT_PROIECTE)
REFERENCES MEMBRI_ACTIVI(ID_MEMBRU),
CONSTRAINT FK_REPREZENTANT_PR FOREIGN KEY(REPREZENTANT_PR)
REFERENCES MEMBRI_ACTIVI(ID_MEMBRU),
CONSTRAINT FK_REPREZENTANT_HR FOREIGN KEY(REPREZENTANT_HR)
REFERENCES MEMBRI_ACTIVI(ID_MEMBRU),
CONSTRAINT FK_REPREZENTANT_MK FOREIGN KEY(REPREZENTANT_MK)
REFERENCES MEMBRI_ACTIVI(ID_MEMBRU));

```

```

ALTER TABLE PROIECT
ADD(ID_CLUB NUMBER(4));

```

```

ALTER TABLE PROIECT
ADD CONSTRAINT FK_PROIECT_CLUBURI
FOREIGN KEY (ID_CLUB)
REFERENCES CLUBURI(ID_CLUB);

```

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID_PROIECT	NUMBER(4, 0)	No	(null)	1	(null)
2	TITLU_PROIECT	VARCHAR2(30 BYTE)	Yes	(null)	2	(null)
3	SCOP	VARCHAR2(20 BYTE)	Yes	(null)	3	(null)
4	DATA_DESFASURARE	DATE	Yes	(null)	4	(null)
5	REPREZENTANT_PROIECTE	NUMBER(4, 0)	Yes	(null)	5	(null)
6	REPREZENTANT_PR	NUMBER(4, 0)	Yes	(null)	6	(null)
7	REPREZENTANT_HR	NUMBER(4, 0)	Yes	(null)	7	(null)
8	REPREZENTANT_MK	NUMBER(4, 0)	Yes	(null)	8	(null)

```

CREATE TABLE TASK
(ID_TASK NUMBER(4),
TITLU_TASK VARCHAR(30) NOT NULL,
STATUS VARCHAR(20),
DEADLINE DATE,
ID_PROIECT NUMBER(4),

```

```

ID_MEMBRU NUMBER(4),
PRIMARY KEY(ID_TASK),
CONSTRAINT FK_ID_PROIECT FOREIGN KEY(ID_PROIECT)
REFERENCES PROIECT(ID_PROIECT),
CONSTRAINT FK_ID_MEMBRU FOREIGN KEY(ID_MEMBRU)
REFERENCES MEMBRI_ACTIVI(ID_MEMBRU)
);

```

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID_TASK	NUMBER(4,0)	No	(null)	1	(null)
2	TITLU_TASK	VARCHAR2(30 BYTE)	No	(null)	2	(null)
3	STATUS	VARCHAR2(20 BYTE)	Yes	(null)	3	(null)
4	DEADLINE	DATE	Yes	(null)	4	(null)
5	ID_PROIECT	NUMBER(4,0)	Yes	(null)	5	(null)
6	ID_MEMBRU	NUMBER(4,0)	Yes	(null)	6	(null)

```

CREATE TABLE PACHET_DE_FINANTARE
(ID_PACHET NUMBER(2),
NUME_PACHET VARCHAR(20),
PROMOVARE_ONLINE VARCHAR(5),
AFISARE_SIGLA VARCHAR(5),
ROLL_UP VARCHAR(5),
DISTRIBUIRE_FLYERE VARCHAR(5),
COMUNICAT_PRESA VARCHAR(5),
PROMOVARE_SITE VARCHAR(5),
INVITARE_REPREZENTANT VARCHAR(5),
ALEGE_PROIECT VARCHAR(5),

PRIMARY KEY(ID_PACHET),
CONSTRAINT U_NUME_PACHET UNIQUE(NUME_PACHET));

```

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PACHET	NUMBER (2, 0)	No	(null)	1	(null)
2 NUME_PACHET	VARCHAR2(20 BYTE)	Yes	(null)	2	(null)
3 PROMOVARE_ONLINE	VARCHAR2(5 BYTE)	Yes	(null)	3	(null)
4 AFISARE_SIGLA	VARCHAR2(5 BYTE)	Yes	(null)	4	(null)
5 ROLL_UP	VARCHAR2(5 BYTE)	Yes	(null)	5	(null)
6 DISTRIBUIRE_FLYERE	VARCHAR2(5 BYTE)	Yes	(null)	6	(null)
7 COMUNICAT_PRESA	VARCHAR2(5 BYTE)	Yes	(null)	7	(null)
8 PROMOVARE_SITE	VARCHAR2(5 BYTE)	Yes	(null)	8	(null)
9 INVITARE_REPREZENTANT	VARCHAR2(5 BYTE)	Yes	(null)	9	(null)
10 ALEGE_PROIECT	VARCHAR2(5 BYTE)	Yes	(null)	10	(null)

CREATE TABLE SPONSORI

```
(ID_SPONSOR NUMBER(4),
NUME_SPONSOR VARCHAR(30),
ID_PACHET NUMBER(2),
SUMA NUMBER(5),
ID_PROIECT NUMBER(4),
```

PRIMARY KEY(ID_SPONSOR),

```
CONSTRAINT FK_ID_PACHET FOREIGN KEY(ID_PACHET)
REFERENCES PACHET_DE_FINANTARE(ID_PACHET),
CONSTRAINT CK_SUMA CHECK(SUMA > 0),
CONSTRAINT FK_ID_PROIECT_SPONSORI FOREIGN KEY(ID_PROIECT)
REFERENCES PROIECT(ID_PROIECT));
```

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_SPONSOR	NUMBER(4, 0)	No	(null)	1	(null)
2 NUME_SPONSOR	VARCHAR2(30 BYTE)	Yes	(null)	2	(null)
3 ID_PACHET	NUMBER(2, 0)	Yes	(null)	3	(null)
4 SUMA	NUMBER(5, 0)	Yes	(null)	4	(null)
5 ID_PROIECT	NUMBER(4, 0)	Yes	(null)	5	(null)

CREATE TABLE FILIALE

```
(ID_FILIALA NUMBER(4),
NUME_FILIALA VARCHAR(30),
```

PRIMARY KEY(ID_FILIALA),

CONSTRAINT U_NUME_FILIALA UNIQUE(NUME_FILIALA));

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_FILIALA	NUMBER(4, 0)	No	(null)	1	(null)
2 NUME_FILIALA	VARCHAR2(30 BYTE)	Yes	(null)	2	(null)

```

CREATE TABLE CLUBURI
(ID_CLUB NUMBER(4),
NUME_CLUB VARCHAR(20),
ID_FILIALA NUMBER(4),

PRIMARY KEY(ID_CLUB),
CONSTRAINT U_NUME_CLUB UNIQUE(NUME_CLUB));

```

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_CLUB	NUMBER(4, 0)	No	(null)	1	(null)
2 NUME_CLUB	VARCHAR2(20 BYTE)	Yes	(null)	2	(null)
3 ID_FILIALA	NUMBER(4, 0)	Yes	(null)	3	(null)

```

CREATE TABLE MEMBRI_ALUMNI
(ID_MEMBRU NUMBER(4),
NUME VARCHAR(20) NOT NULL,
PRENUME VARCHAR(20),
EMAIL CHAR(30),
NR_TEL VARCHAR(15),
DATA_ADERARE DATE,
DATA_RETTRAGERE DATE,

PRIMARY KEY(ID_MEMBRU),
CONSTRAINT U_NUME_PRENUME_ALUMNI UNIQUE(NUME, PRENUME),
CONSTRAINT U_EMAIL_ALUMNI UNIQUE(EMAIL));

```

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID_MEMBRU	NUMBER(4,0)	No	(null)	1	(null)
2	NUME	VARCHAR2(20 BYTE)	No	(null)	2	(null)
3	PRENUME	VARCHAR2(20 BYTE)	Yes	(null)	3	(null)
4	EMAIL	CHAR(30 BYTE)	Yes	(null)	4	(null)
5	NR_TEL	VARCHAR2(15 BYTE)	Yes	(null)	5	(null)
6	DATA_ADERARE	DATE	Yes	(null)	6	(null)
7	DATA_RETIRAGERE	DATE	Yes	(null)	7	(null)

ALTER TABLE DEPARTAMENTE
ADD CONSTRAINT FK_CLUB_ID FOREIGN KEY(ID_CLUB)
REFERENCES CLUBURI(ID_CLUB);

5. Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).

INSERT INTO FILIALE
VALUES(SEQ.NEXTVAL, 'Iasi');

	ID_FILIALA	NUME_FILIALA
1	1	Iasi

INSERT INTO CLUBURI
VALUES(SEQ.NEXTVAL, 'Iasi', 1);

INSERT INTO CLUBURI
VALUES(SEQ.NEXTVAL, 'Vaslui', 1);

INSERT INTO CLUBURI
VALUES(SEQ.NEXTVAL, 'Roman', 1);

	ID_CLUB	NUME_CLUB	ID_FILIALA
1	2	Iasi	1
2	3	Vaslui	1
3	4	Roman	1

```
INSERT INTO DEPARTAMENTE
VALUES(SEQ.NEXTVAL, 'Scriere de Proiecte', 10, 2);
```

```
INSERT INTO DEPARTAMENTE
VALUES(SEQ.NEXTVAL, 'Relatii Publice', 11, 2);
```

```
INSERT INTO DEPARTAMENTE
VALUES(SEQ.NEXTVAL, 'Resurse Umane', 12, 2);
```

```
INSERT INTO DEPARTAMENTE
VALUES(SEQ.NEXTVAL, 'Marketing', 13, 2);
```

```
INSERT INTO DEPARTAMENTE
VALUES(SEQ.NEXTVAL, 'Consiliu Executiv', 14, 2);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,
ID_DEPARTAMENT)
VALUES(SEQ.NEXTVAL, 'Baciu', 'Marius', '13-JUN-21', 5);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,
ID_DEPARTAMENT)
VALUES(SEQ.NEXTVAL, 'Craciun', 'Dana', '16-MAR-22', 6);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,
ID_DEPARTAMENT)
VALUES(SEQ.NEXTVAL, 'Cirlan', 'Denisa', '22-JAN-21', 7);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,
ID_DEPARTAMENT)
VALUES(SEQ.NEXTVAL, 'Ion', 'Marian', '17-MAR-22', 8);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)  
VALUES(SEQ.NEXTVAL, 'Popescu', 'Sara', '10-MAY-21', 9);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)  
VALUES(SEQ.NEXTVAL, 'Vlad', 'Andreea', '16-APR-23', 5);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)  
VALUES(SEQ.NEXTVAL, 'Dinu', 'Tudor', '22-JUN-23', 6);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)  
VALUES(SEQ.NEXTVAL, 'Pinzaru', 'Alex', '23-JUL-23', 7);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)  
VALUES(SEQ.NEXTVAL, 'Gologan', 'Ianis', '13-DEC-22', 8);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)  
VALUES(SEQ.NEXTVAL, 'Grigore', 'Teodora', '19-MAY-20', 9);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)  
VALUES(SEQ.NEXTVAL, 'Valeanu', 'Teodora', '22-JAN-23', 5);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)  
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)  
VALUES(SEQ.NEXTVAL, 'Perju', 'Andreea', '23-JUL-23', 7);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)
```

```
VALUES(SEQ.NEXTVAL, 'Gheorghe', 'Flavia', '12-JUL-22', 8);
```

```
INSERT INTO MEMBRI_ACTIVI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
ID_DEPARTAMENT)
```

```
VALUES(SEQ.NEXTVAL,'Feraru','Catalina','22-MAR-20',9);
```

```
INSERT INTO MEMBRI_ALUMNI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
DATA_RETTRAGERE)
```

```
VALUES(SEQ.NEXTVAL, 'Anton', 'Florin', '14-APR-19', '16-SEP-22');
```

```
INSERT INTO MEMBRI_ALUMNI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
DATA_RETIRAGERE)
```

```
VALUES(SEQ.NEXTVAL, 'Gheorghe', 'Andrei', '14-MAY-18', '17-OCT-21');
```

```
INSERT INTO MEMBRI_ALUMNI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
DATA_RETIRAGERE)
```

```
INSERT INTO MEMBRI_ALUMNI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
DATA_RETIRAGERE)
```

```
VALUES(SEQ.NEXTVAL, 'Dan', 'Gicu', '4-Feb-20', '28-SEP-22');
```

```
INSERT INTO MEMBRI_ALUMNI(ID_MEMBRU, NUME, PRENUME, DATA_ADERARE,  
DATA_RETIRARE)
```

```
VALUES(SEQ.NEXTVAL,'Alexe','Mihai','14-APR-20','8-MAY-22');
```

ID_MEMBRU	NUME	PRENUME	EMAIL	NR_TEL	DATAADERARE	DATA_RETRAGERE	
1	41	Anton	Florin	(null)	(null)	14-APR-19	16-SEP-22
2	42	Gheorghe	Andrei	(null)	(null)	14-MAY-18	17-OCT-21
3	43	Penisoara	Gelu	(null)	(null)	19-APR-19	20-OCT-22
4	44	Dan	Gicu	(null)	(null)	04-FEB-20	28-SEP-22
5	45	Alexe	Mihai	(null)	(null)	14-APR-20	08-MAY-22

	ID_MEMBRU	NUME	PRENUME	EMAIL	NR_TEL	DATA_ADERARE	ID_DEPARTAMENT
1	10	Baciu	Marius	(null)	(null)	13-JUN-21	5
2	11	Craciun	Dana	(null)	(null)	16-MAR-22	6
3	12	Cirlan	Denisa	(null)	(null)	22-JAN-21	7
4	13	Ion	Marian	(null)	(null)	17-MAR-22	8
5	14	Popescu	Sara	(null)	(null)	10-MAY-21	9
6	15	Vlad	Andreea	(null)	(null)	16-APR-23	5
7	16	Dinu	Tudor	(null)	(null)	22-JUN-23	6
8	17	Pinzaru	Alex	(null)	(null)	23-JUL-23	7
9	18	Gologan	Ianis	(null)	(null)	13-DEC-22	8
10	19	Grigore	Teodora	(null)	(null)	19-MAY-20	9
11	21	Valeanu	Teodora	(null)	(null)	22-JAN-23	5
12	22	Penisoara	Tudor	(null)	(null)	18-MAY-19	6
13	23	Pinzaru	Andreea	(null)	(null)	23-JUL-23	7
14	24	Negara	Daniel	(null)	(null)	12-JUL-22	8
15	25	Florea	Dragos	(null)	(null)	22-MAR-20	9
16	26	Maftei	Ernest	(null)	(null)	22-JAN-23	5
17	27	Iordan	Tristana	(null)	(null)	18-MAY-19	6
18	28	Perju	Andreea	(null)	(null)	23-JUL-23	7
19	29	Gheorghe	Flavia	(null)	(null)	12-JUL-22	8
20	30	Feraru	Catalina	(null)	(null)	22-MAR-20	9
21	89	Mocanu	Andrei	(null)	(null)	14-APR-21	84
22	90	Mocanu	Matei	(null)	(null)	12-JAN-22	85
23	91	Praslea	Edesia	(null)	(null)	12-JAN-21	86
24	92	Paslaru	Marisa	(null)	(null)	03-MAR-22	87
25	93	Spuma	Ioana	(null)	(null)	10-NOV-21	88
26	94	Boicu	Ionut	(null)	(null)	16-APR-20	84
27	95	Amaritei	Irina	(null)	(null)	22-JUN-21	85
28	96	Arhire	Anemona	(null)	(null)	23-DEC-23	86
29	97	Popa	Denis	(null)	(null)	13-DEC-22	87
30	98	Mihalcea	Mihai	(null)	(null)	10-MAY-20	88

31	99	Alexe	Alex	(null)	(null)	18-JAN-23		84
32	100	Badac	Maria	(null)	(null)	02-MAY-19		85
33	101	Streja	Sandica	(null)	(null)	23-JUL-23		86
34	102	Bradea	Codrin	(null)	(null)	12-JUL-22		87
35	103	Badea	Ovidiu	(null)	(null)	04-MAR-20		88
36	104	Balu	Roberta	(null)	(null)	12-JAN-21		84
37	105	Bumbu	Raluca	(null)	(null)	13-MAY-19		85
38	106	Nimirceag	Loredana	(null)	(null)	22-APR-23		86
39	107	Gheorghe	Lucian	(null)	(null)	12-OCT-22		87
40	108	Lungu	Rares	(null)	(null)	19-MAR-21		88
41	117	Boicu	Marius	(null)	(null)	04-MAY-21		112
42	118	Mititelu	Maria	(null)	(null)	11-FEB-22		113
43	119	Turcu	Raisa	(null)	(null)	12-JAN-20		114
44	120	Rusu	Maria	(null)	(null)	23-MAR-22		115
45	121	Cirlan	Andreea	(null)	(null)	19-NOV-21		116
46	122	Mandru	Ioana	(null)	(null)	06-MAR-20		112
47	123	Mandache	Tudor	(null)	(null)	23-JUL-21		113
48	124	Munteanu	Alex	(null)	(null)	23-SEP-23		114
49	125	Munteanu	Tudor	(null)	(null)	12-DEC-22		115
50	126	Hulubei	Gabriel	(null)	(null)	09-MAY-20		116
51	127	Ciuraru	Cosmin	(null)	(null)	18-JAN-23		112
52	128	Mihaila	Cosmin	(null)	(null)	12-MAY-19		113
53	129	Rosu	Paul	(null)	(null)	23-JUL-23		114
54	130	Bot	George	(null)	(null)	13-JUL-22		115
55	131	Ciovnicu	Gina	(null)	(null)	04-APR-20		116
56	132	Carp	Rafael	(null)	(null)	14-JAN-21		112
57	133	Miron	Andra	(null)	(null)	13-MAY-19		113
58	134	Enache	Alexandra	(null)	(null)	22-APR-23		114
59	141	Turcu	Edi	(null)	(null)	12-OCT-22		115

INSERT INTO PROIECT

VALUES(SEQ.NEXTVAL, 'Cariere', 'Informativ', '22-OCT-23', 117, 118, 119, 120,4);

INSERT INTO PROIECT

VALUES(SEQ.NEXTVAL, 'Job Almanac', 'Informativ', '23-APR-23', 122, 123, 124, 125,4);

INSERT INTO PROIECT

VALUES(SEQ.NEXTVAL, 'Arta n Dar', 'Caritabil', '20-DEC-23', 127, 128, 129, 130,4);

INSERT INTO PROIECT

VALUES(SEQ.NEXTVAL, 'RomanMUN', 'Academic', '6-AUG-23', 132, 133, 134, 141,4);

ID_PROIECT	TITLU_PROJECT	SCOP	DATA_DESFASURARE	REPREZENTANT_PROIECTE	REPREZENTANT_PR	REPREZENTANT_HR	REPREZENTANT_MK
1	61 ISMUN	Academic	16-AUG-23	15	16	17	18
2	82 Viata pe Bune	Informativ	06-JUL-23	21	22	23	24
3	83 Shelter Stories	Caritabil	23-MAR-23	26	27	28	29
4	109 Christmas Groove	Caritabil	20-DEC-23	94	95	96	97
5	110 Bonding Time	Social	19-SEP-23	99	100	101	102
6	111 Movie Night	Caritabil	19-JUN-23	104	105	106	107
7	143 Cariere	Informativ	22-OCT-23	117	118	119	120
8	144 Job Almanac	Informativ	23-APR-23	122	123	124	125
9	145 Arta n Dar	Caritabil	20-DEC-23	127	128	129	130
10	146 RomanMUN	Academic	06-AUG-23	132	133	134	141

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'FabLab Iasi', 3, 3000, 61);

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'Plantagro', 3, 2000, 146);

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'Adobe', 2, 2000, 143);

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'Bitdefender', 2, 1500, 143);

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'Hiccup', 3, 3000, 83);

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'Miaunel', 1, 500, 82);

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'Blanca', 1, 300, 82);

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'Centrul de Afaceri', 3, 4000, 109);

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'Rotaru', 2, 3000, 145);

INSERT INTO SPONSORI

```

VALUES(SEQ.NEXTVAL, 'Hiccup', 2, 2000, 111);

INSERT INTO SPONSORI
VALUES(SEQ.NEXTVAL, 'OpportuniTool', 3, 2500, 144);

```

	ID_SPONSOR	NUME_SPONSOR	ID_PACHET	SUMA	ID_PROIECT
1	161	FabLab Iasi	3	3000	61
2	162	Plantagro	3	2000	146
3	163	Adobe	2	2000	143
4	164	Bitdefender	2	1500	143
5	165	Hiccup	3	3000	83
6	166	Miaunel	1	500	82
7	167	Blanca	1	300	82
8	168	Centrul de Afaceri	3	4000	109
9	169	Rotaru	2	3000	145
10	170	Hiccup	2	2000	111
11	171	OpportuniTool	3	2500	144

```

INSERT INTO TASK
VALUES(SEQ.NEXTVAL, 'Formular Inscriere', 'Realizat', '5-AUG-23', 61, 17);

```

```

INSERT INTO TASK
VALUES(SEQ.NEXTVAL, 'Cautat Joculete', 'In curs', '5-SEP-23', 110, 101);

```

```

INSERT INTO TASK
VALUES(SEQ.NEXTVAL, 'Contactat Speakeri', 'Realizat', '5-JUL-23', 82, 24);

```

```

INSERT INTO TASK
VALUES(SEQ.NEXTVAL, 'Contactat Speakeri', 'Realizat', '5-MAR-23', 83, 29);

```

```

INSERT INTO TASK
VALUES(SEQ.NEXTVAL, 'Cautat Sala', 'Neinceput', '5-DEC-23', 109, 97);

```

```

INSERT INTO TASK
VALUES(SEQ.NEXTVAL, 'Cautat Film', 'Realizat', '5-JUN-23', 111, 104);

```

```
INSERT INTO TASK
```

```
VALUES(SEQ.NEXTVAL, 'Mapa de Proiect', 'Realizat', '5-OCT-23', 143, 117);
```

```
INSERT INTO TASK
```

```
VALUES(SEQ.NEXTVAL, 'Cautat Speakeri', 'Realizat', '5-APR-23', 144, 122);
```

```
INSERT INTO TASK
```

```
VALUES(SEQ.NEXTVAL, 'Cautat Colinde', 'Neinceput', '5-DEC-23', 145, 127);
```

```
INSERT INTO TASK
```

```
VALUES(SEQ.NEXTVAL, 'Contactat Scoala', 'Realizat', '5-AUG-23', 146, 141);
```

ID_TASK	TITLU_TASK	STATUS	DEADLINE	ID_PROIECT	ID_MEMBRU
1	181 Formular Inscriere	Realizat	05-AUG-23	61	17
2	182 Cautat Jocuite	In curs	05-SEP-23	110	101
3	183 Contactat Speakeri	Realizat	05-JUL-23	82	24
4	184 Contactat Speakeri	Realizat	05-MAR-23	83	29
5	185 Cautat Sala	Neinceput	05-DEC-23	109	97
6	187 Mapa de Proiect	Realizat	05-OCT-23	143	117
7	188 Cautat Film	Realizat	05-JUN-23	111	104
8	189 Cautat Speakeri	Realizat	05-APR-23	144	122
9	190 Cautat Colinde	Neinceput	05-DEC-23	145	127
10	191 Contactat Scoala	Realizat	05-AUG-23	146	141

6. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze toate cele 3 tipuri de colecții studiate. Apelați subprogramul.

Cerință: Să se creeze: -un vector care să rețină ID-urile tuturor taskurilor care nu au fost atribuite vreunui membru;

- un tablou imbricat care să rețină ID-urile tuturor membrilor cărora le-au fost atribuite task-uri;
- un tablou indexat care să rețină toate taskurile finalizate. Inițial să se șteargă aceste taskuri.

Apoi, să li se atribuie taskurile neatribuite, membrilor fără taskuri.

```

CREATE OR REPLACE PROCEDURE atribuire_taskuri
IS
    statusul TASK.STATUS%TYPE;

    TYPE TASK_TABLE IS TABLE OF TASK%ROWTYPE INDEX BY PLS_INTEGER;
    TYPE TASK_VARRAY IS VARRAY(100) OF TASK%ROWTYPE;
    TYPE TASK_MEMBRI IS TABLE OF TASK.ID_MEMBRU%TYPE;

    contor NUMBER := 1;
    taskuri_realizate TASK_TABLE;
    taskuri_fara_membru TASK_VARRAY := TASK_VARRAY();
    id_membri_asignati TASK_MEMBRI := TASK_MEMBRI();

    CURSOR c_taskuri_realizate IS
        SELECT *
        FROM TASK
        WHERE STATUS = 'Realizat';

    CURSOR c_taskuri_fara_membru IS
        SELECT *
        FROM TASK
        WHERE ID_MEMBRU IS NULL;

    CURSOR c_taskuri_cu_membru IS
        SELECT DISTINCT ID_MEMBRU
        FROM TASK
        WHERE ID_MEMBRU IS NOT NULL;

    CURSOR c_membri_activi IS
        SELECT ID_MEMBRU
        FROM MEMBRI_ACTIVI;

BEGIN
    FOR task_rec IN c_taskuri_realizate LOOP
        taskuri_realizate(taskuri_realizate.COUNT + 1) := task_rec;
    END LOOP;

```

```

FOR i IN 1..taskuri_realizate.COUNT LOOP
    DELETE
    FROM TASK
    WHERE ID_TASK = taskuri_realizate(i).ID_TASK;
END LOOP;

FOR task_rec IN c_taskuri_fara_membru LOOP
    taskuri_fara_membru.EXTEND;
    taskuri_fara_membru(taskuri_fara_membru.LAST) := task_rec;
END LOOP;

FOR task_rec IN c_taskuri_cu_membru LOOP
    id_membri_asignati.EXTEND;
    id_membri_asignati(id_membri_asignati.LAST) := task_rec.ID_MEMBRU;
END LOOP;

FOR membru_rec IN c_membri_activi LOOP
    IF contor <= taskuri_fara_membru.COUNT THEN
        UPDATE TASK
        SET ID_MEMBRU = membru_rec.ID_MEMBRU
        WHERE ID_TASK = taskuri_fara_membru(contor).ID_TASK;
        contor := contor + 1;
    ELSE
        EXIT;
    END IF;
END LOOP;
END atribuire_taskuri;
/

```

```

BEGIN
    atribuire_taskuri;
END;
/

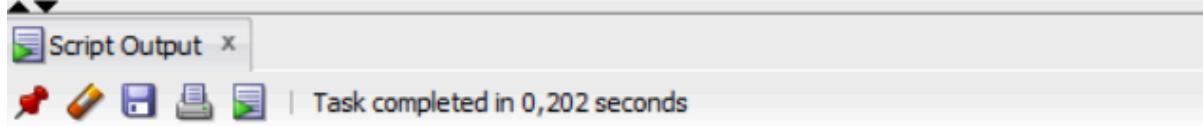
```

	ID_TASK	TITLU_TASK	STATUS	DEADLINE	ID_PROIECT	ID_MEMBRU
1	182	Cautat Joculete	In curs	05-SEP-23	110	101
2	185	Cautat Sala	Neinceput	05-DEC-23	109	97
3	190	Cautat Colinde	Neinceput	05-DEC-23	145	127
4	192	Formular Chairs	Neinceput	07-AUG-23	61	(null)
5	193	Contact Locatii	Neinceput	08-AUG-23	61	(null)
6	194	Diplome	Neinceput	09-AUG-23	61	(null)
7	195	Contact Brunch	Neinceput	10-AUG-23	61	(null)
8	196	Formular Chairs	Neinceput	07-AUG-23	61	407
9	197	Formular Chairs	Neinceput	07-AUG-23	61	407
10	198	Formular Chairs	Neinceput	07-AUG-23	61	407

```

1 CREATE OR REPLACE PROCEDURE atribuire_taskuri
2   IS
3     statusul TASK.STATUS%TYPE;
4
5     TYPE TASK_TABLE IS TABLE OF TASK%ROWTYPE INDEX BY PLS_INTEGER;
6     TYPE TASK_VARRAY IS VARRAY(100) OF TASK%ROWTYPE;
7     TYPE TASK_MEMBRI IS TABLE OF TASK.ID_MEMBRU%TYPE;
8
9     contor NUMBER := 1;
10    taskuri_realizate TASK_TABLE;
11    taskuri_fara_membru TASK_VARRAY := TASK_VARRAY();
12    id_membri_asignati TASK_MEMBRI := TASK_MEMBRI();
13
14    CURSOR c_taskuri_realizate IS
15      SELECT *
16        FROM TASK
17       WHERE STATUS = 'Realizat';
18
19    CURSOR c_taskuri_fara_membru IS
20      SELECT *

```



Procedure ATRIBUIRE_TASKURI compiled

PL/SQL procedure successfully completed.

ID_TASK	TITLU_TASK	STATUS	DEADLINE	ID_PROIECT	ID_MEMBRU
1	182 Cautat Joclete	In curs	05-SEP-23	110	101
2	185 Cautat Sala	Neinceput	05-DEC-23	109	97
3	190 Cautat Colinde	Neinceput	05-DEC-23	145	127
4	192 Formular Chairs	Neinceput	07-AUG-23	61	10
5	193 Contact Locatii	Neinceput	08-AUG-23	61	11
6	194 Diplome	Neinceput	09-AUG-23	61	12
7	195 Contact Brunch	Neinceput	10-AUG-23	61	13
8	196 Formular Chairs	Neinceput	07-AUG-23	61	407
9	197 Formular Chairs	Neinceput	07-AUG-23	61	407
10	198 Formular Chairs	Neinceput	07-AUG-23	61	407

7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat, dependent de celălalt cursor. Apelați subprogramul.

Cerință: Pentru fiecare membru, să se afișeze lista taskurilor.

```
CREATE OR REPLACE PROCEDURE afisare_taskuri
```

```
IS
```

```
v_id_membru TASK.ID_MEMBRU%TYPE;
```

```
CURSOR c_membri_activi IS
```

```
SELECT ID_MEMBRU
```

```
FROM MEMBRI_ACTIVI;
```

```
CURSOR c_taskuri_membri(p_id_membru TASK.ID_MEMBRU%TYPE) IS
```

```
SELECT *
```

```
FROM TASK
```

```
WHERE ID_MEMBRU = p_id_membru;
```

```
BEGIN
```

```
FOR membri_rec IN c_membri_activi LOOP
```

```
    v_id_membru := membri_rec.ID_MEMBRU;
```

```
    DBMS_OUTPUT.PUT_LINE('ID_MEMBRU: ' || v_id_membru);
```

```

FOR task_rec IN c_taskuri_membri(v_id_membru) LOOP
    DBMS_OUTPUT.PUT_LINE(' TASK ID:' || task_rec.ID_TASK || ', Status:' ||
task_rec.STATUS);
END LOOP;
DBMS_OUTPUT.PUT_LINE('-----');
END LOOP;

END afisare_taskuri;
/

```

```

BEGIN
    afisare_taskuri;
END;
/

```

The screenshot shows the Oracle SQL Developer interface with the following details:

- Worksheet Tab:** Contains the PL/SQL code for the procedure `afisare_taskuri`.
- Script Output Tab:** Shows the output of the compilation command: `Procedure AFISARE_TASKURI compiled` and `PL/SQL procedure successfully completed.`
- Dbms Output Tab:** Displays the output of the `DBMS_OUTPUT.PUT_LINE` statements, listing members with their task IDs and statuses. The output is as follows:

ID_MEMBRU	TASK ID	Status
10	192	Neinceput
11	193	Neinceput
12	194	Neinceput
13	195	Neinceput
14		
15		
16		
17		
18		
19		
20		
21		
22		
23		

8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 exceptii proprii. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate.

Cerință: Să se afișeze pentru un proiect cu ID-ul dat ca parametru, bugetul proiectului după ce se scade suma de 1500 RON, pentru cumpărarea materialelor.

Excepții: -proiectul are mai puțin de 300 RON alocăți (este proiect intern);
-proiectul nu are alocăți suficienți bani.

```
CREATE OR REPLACE FUNCTION actualizare_buget
(p_id_proiect PROIECT.ID_PROIECT%TYPE)
RETURN SPONSORI.SUMA%TYPE IS
    buget SPONSORI.SUMA%TYPE;
    v_titlu_proiect PROIECT.TITLU_PROIECT%TYPE;
    v_nume_club CLUBURI.NUME_CLUB%TYPE;

    PROIECT_INTERN EXCEPTION;
    BUGET_INSUFICIENT EXCEPTION;

BEGIN
    SELECT
        SUM(S.SUMA) - 1500, P.TITLU_PROIECT, C.NUME_CLUB
    INTO buget, v_titlu_proiect, v_nume_club
    FROM SPONSORI S
    JOIN PROIECT P
    ON(S.ID_PROIECT = P.ID_PROIECT)
    JOIN CLUBURI C
    ON(P.ID_CLUB = C.ID_CLUB)
    WHERE S.ID_PROIECT = p_id_proiect
    GROUP BY P.TITLU_PROIECT, C.NUME_CLUB;

    IF buget + 1500 <= 300 THEN
        RAISE PROIECT_INTERN;
    END IF;

    IF buget < 0 THEN
        RAISE BUGET_INSUFICIENT;
    END IF;
```

```
DBMS_OUTPUT.PUT_LINE('Proiectul ' || v_titlu_proiect || ' din clubul ' ||  
v_nume_club || ' va ramane cu un buget de ');\n    RETURN buget;\n\nEXCEPTION\n    WHEN NO_DATA_FOUND OR PROIECT_INTERN THEN\n        RAISE_APPLICATION_ERROR(-20004, 'Proiectul este intern si nu are nevoie de  
materiale!');\n\n    WHEN BUGET_INSUFICIENT THEN\n        RAISE_APPLICATION_ERROR(-20003, 'Bugetul este prea mic pentru realizarea  
cumparaturilor!');\n\nEND actualizare_buget;\n/\n\nBEGIN\n    DBMS_OUTPUT.PUT_LINE(actualizare_buget(61));\nEND;\n/\n\nBEGIN\n    DBMS_OUTPUT.PUT_LINE(actualizare_buget(110));\nEND;\n/\n\nBEGIN\n    DBMS_OUTPUT.PUT_LINE(actualizare_buget(82));\nEND;\n/
```

Dbms Output

+ | Buffer Size: 20000 |

docker_proiect *

```
Proiectul ISMUN din clubul Iasi va ramane cu un buget de  
1500
```

```
Error starting at line : 48 in command -  
BEGIN  
    DBMS_OUTPUT.PUT_LINE(actualizare_buget(110));  
END;  
Error report -  
ORA-20004: Proiectul este intern si nu are nevoie de materiale!  
ORA-06512: at "PROIECT.ACTUALIZARE_BUGET", line 35  
ORA-06512: at line 2
```

```
Error starting at line : 53 in command -  
BEGIN  
    DBMS_OUTPUT.PUT_LINE(actualizare_buget(82));  
END;  
Error report -  
ORA-20003: Bugetul este prea mic pentru realizarea cumparaturilor!  
ORA-06512: at "PROIECT.ACTUALIZARE_BUGET", line 38  
ORA-06512: at line 2
```

9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO_DATA_FOUND și TOO_MANY_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Cerință: Să se afișeze cel mai activ membru dintre toate cluburile.

Punctajul se va calcula astfel:

- 1 punct pentru fiecare task atribuit;
- 2 puncte pentru fiecare funcție de reprezentant la un proiect.

Excepții: -NO_DATA_FOUND: nu există niciun membru activ;

-TOO_MANY_ROWS: sunt mai mulți voluntari activi cu punctaj maxim.

```
CREATE OR REPLACE PROCEDURE cel_mai_activ_membru
IS
    v_id_membru MEMBRI_ACTIVI.ID_MEMBRU%TYPE;
    v_punctaj NUMBER;
BEGIN
    SELECT ID_MEMBRU, punctaj INTO v_id_membru, v_punctaj
    FROM (
        SELECT M.ID_MEMBRU,
               COUNT(T.ID_MEMBRU) + 2 * COUNT(P.ID_COORDONATOR) AS punctaj,
               RANK() OVER (ORDER BY COUNT(T.ID_MEMBRU) + 2 *
                             COUNT(P.ID_COORDONATOR) DESC) AS rnk
        FROM CLUBURI C
        JOIN DEPARTAMENTE D ON (C.ID_CLUB = D.ID_CLUB)
        JOIN MEMBRI_ACTIVI M ON (D.ID_DEPARTAMENT = M.ID_DEPARTAMENT)
        LEFT JOIN TASK T ON (M.ID_MEMBRU = T.ID_MEMBRU)
        LEFT JOIN PROIECT P ON (M.ID_MEMBRU = P.ID_COORDONATOR)
        GROUP BY M.ID_MEMBRU
    ) WHERE rnk = 1;

    DBMS_OUTPUT.PUT_LINE('ID Membru cu punctaj maxim: ' || v_id_membru || ',';
    Punctaj: ' || v_punctaj);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20005, 'Un departament nu are membri.');
    WHEN TOO_MANY_ROWS THEN
        RAISE_APPLICATION_ERROR(-20006, 'Un departament are mai multi membri
        cu acelasi punctaj.');
END cel_mai_activ_membru;
/
BEGIN
    cel_mai_activ_membru;
END;
```

```

1 CREATE OR REPLACE PROCEDURE cel_mai_activ_membru
2 IS
3     v_id_membru MEMBRI_ACTIVI.ID_MEMBRU%TYPE;
4     v_punctaj NUMBER;
5 BEGIN
6     SELECT ID_MEMBRU, punctaj INTO v_id_membru, v_punctaj
7     FROM (
8         SELECT M.ID_MEMBRU,
9             COUNT(T.ID_MEMBRU) + 2 * COUNT(P.ID_COORDONATOR) AS punctaj,
10            RANK() OVER (ORDER BY COUNT(T.ID_MEMBRU) + 2 * COUNT(P.ID_COORDONATOR) DESC) AS rnk
11        FROM CLUBURI C
12        JOIN DEPARTAMENTE D ON (C.ID_CLUB = D.ID_CLUB)
13        JOIN MEMBRI_ACTIVI M ON (D.ID_DEPARTAMENT = M.ID_DEPARTAMENT)
14        LEFT JOIN TASK T ON (M.ID_MEMBRU = T.ID_MEMBRU)
15        LEFT JOIN PROIECT P ON (M.ID_MEMBRU = P.ID_COORDONATOR)
16        GROUP BY M.ID_MEMBRU
17    ) WHERE rnk = 1;
18
19    DBMS_OUTPUT.PUT_LINE('ID Membru cu punctaj maxim: ' || v_id_membru || ', Punctaj: ' || v_punctaj);
20 EXCEPTION
21     WHEN NO_DATA_FOUND THEN
22         RAISE_APPLICATION_ERROR(-20005, 'Un departament nu are membri.');
23     WHEN TOO_MANY_ROWS THEN
24         RAISE_APPLICATION_ERROR(-20006, 'Un departament are mai multi membri cu acelasi punctaj.');
25 END cel_mai_activ_membru;
26 /
27

```

	ID_TASK	TITLU_TASK	STATUS	DEADLINE	ID_PROIECT	ID_MEMBRU
1	182	Cautat Joculete	In curs	05-SEP-23	110	101
2	185	Cautat Sala	Neinceput	05-DEC-23	109	97
3	190	Cautat Colinde	Neinceput	05-DEC-23	145	127
4	192	Formular Chairs	Neinceput	07-AUG-23	61	10
5	193	Contact Locatii	Neinceput	08-AUG-23	61	11
6	194	Diplome	Neinceput	09-AUG-23	61	12
7	195	Contact Brunch	Neinceput	10-AUG-23	61	13

PL/SQL procedure successfully completed.

Dbms Output

docker_project x

```
ID Membru cu punctaj maxim: 127, Punctaj: 3
```

ID_TASK	TITLU_TASK	STATUS	DEADLINE	ID_PROIECT	ID_MEMBRU
1	182 Cautat Jocuite	In curs	05-SEP-23	110	101
2	185 Cautat Sala	Neinceput	05-DEC-23	109	97
3	190 Cautat Colinde	Neinceput	05-DEC-23	145	127
4	192 Formular Chairs	Neinceput	07-AUG-23	61	10
5	193 Contact Locatii	Neinceput	08-AUG-23	61	11
6	194 Diplome	Neinceput	09-AUG-23	61	12
7	195 Contact Brunch	Neinceput	10-AUG-23	61	13
8	196 Formular Chairs	Neinceput	07-AUG-23	61	407
9	197 Formular Chairs	Neinceput	07-AUG-23	61	407
10	198 Formular Chairs	Neinceput	07-AUG-23	61	407

Procedure CEL_MAI_ACTIV_MEMBRU compiled

```
Error starting at line : 28 in command -
BEGIN
    cel_mai_activ_membru;
END;
Error report -
ORA-20006: Un departament are mai multi membri cu acelasi punctaj.
ORA-06512: at "PROIECT.CEL_MAI_ACTIV_MEMBRU", line 24
ORA-06512: at line 2
```

10. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul.

Cerință: Să nu fie permise ștergerea și actualizarea tabelului MEMBRI_ALUMNI.

```

CREATE OR REPLACE TRIGGER modificari_alumni
  BEFORE DELETE OR UPDATE ON MEMBRI_ALUMNI
BEGIN
  RAISE_APPLICATION_ERROR(-20005, 'Tabelul MEMBRI_ALUMNI nu poate fi
modificat');
END;
/

```

The screenshot shows the Oracle SQL Developer interface. In the top pane, a code editor displays the trigger definition:

```

1 CREATE OR REPLACE TRIGGER modificari_alumni
2   BEFORE DELETE OR UPDATE ON MEMBRI_ALUMNI
3 BEGIN
4   RAISE_APPLICATION_ERROR(-20005, 'Tabelul MEMBRI_ALUMNI nu poate fi
5 modificat');
6 END;
7 /
8
9 DELETE
10 FROM MEMBRI_ALUMNI
11 WHERE ID_MEMBRU = 46;

```

In the bottom pane, the "Script Output" tab shows the results of running the trigger:

Trigger MODIFICARI_ALUMNI compiled

Error starting at line : 8 in command -
DELETE
FROM MEMBRI_ALUMNI
WHERE ID_MEMBRU = 46
Error at Command Line : 9 Column : 6
Error report -
SQL Error: ORA-20005: Tabelul MEMBRI_ALUMNI nu poate fi modificat
ORA-06512: at "PROIECT.MODIFICARI_ALUMNI", line 2
ORA-04088: error during execution of trigger 'PROIECT.MODIFICARI_ALUMNI'

11. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.

Cerință: Odată cu ștergerea unui membru din tabelul MEMBRI_ACTIVI, acesta să fie adăugat în tabelul MEMBRI_ALUMNI automat.

```

CREATE OR REPLACE TRIGGER actualizare_alumni
  BEFORE DELETE ON MEMBRI_ACTIVI
  FOR EACH ROW

```

```

BEGIN
  INSERT INTO MEMBRI_ALUMNI
  VALUES(SEQ.NEXTVAL, :OLD.NUME, :OLD.PRENUME, :OLD.EMAIL, :OLD.NR_TEL,
:OLD.DATA_ADERARE, SYSDATE);
END actualizare_alumni;
/

```

	ID_MEMBRU	NUME	PRENUME	EMAIL	NR_TEL	DATA_ADERARE	DATA_RETTRAGERE
1	41	Anton	Florin	(null)	(null)	14-APR-19	16-SEP-22
2	42	Gheorghe	Andrei	(null)	(null)	14-MAY-18	17-OCT-21
3	43	Penisoara	Gelu	(null)	(null)	19-APR-19	20-OCT-22
4	44	Dan	Gicu	(null)	(null)	04-FEB-20	28-SEP-22
5	45	Alexe	Mihai	(null)	(null)	14-APR-20	08-MAY-22
6	46	Florescu	(null)	(null)	(null)	(null)	(null)
7	202	Dragos	Stefan	(null)	(null)	26-OCT-23	12-JAN-24

```

1 CREATE OR REPLACE TRIGGER actualizare_alumni
2   BEFORE DELETE ON MEMBRI_ACTIVI
3   FOR EACH ROW
4 BEGIN
5   INSERT INTO MEMBRI_ALUMNI
6   VALUES(SEQ.NEXTVAL, :OLD.NUME, :OLD.PRENUME, :OLD.EMAIL, :OLD.NR_TEL, :OLD.DATA_ADERARE, SYSDATE);
7 END actualizare_alumni;
8 /
9
10 DELETE FROM MEMBRI_ACTIVI
11 WHERE ID_MEMBRU = 408;
12

```

Script Output x

Task completed in 0,02 seconds

Trigger ACTUALIZARE_ALUMNI compiled

1 row deleted.

	ID_MEMBRU	NUME	PRENUME	EMAIL	NR_TEL	DATA_ADERARE	DATA_RETTRAGERE
1	41	Anton	Florin	(null)	(null)	14-APR-19	16-SEP-22
2	42	Gheorghe	Andrei	(null)	(null)	14-MAY-18	17-OCT-21
3	43	Penisoara	Gelu	(null)	(null)	19-APR-19	20-OCT-22
4	44	Dan	Gicu	(null)	(null)	04-FEB-20	28-SEP-22
5	45	Alexe	Mihai	(null)	(null)	14-APR-20	08-MAY-22
6	46	Florescu	(null)	(null)	(null)	(null)	(null)
7	202	Dragos	Stefan	(null)	(null)	26-OCT-23	12-JAN-24
8	201	Craciunescu	Alexandru	(null)	(null)	26-OCT-23	10-JAN-24

12. Definiți un trigger de tip LDD. Declanșați trigger-ul.

Cerință: Să se salveze în tabelul COMENZI cu atributele:

UTILIZATOR VARCHAR2(30),

NUME_BD VARCHAR2(50),

EVENIMENT VARCHAR2(20),

NUME_OBIECT VARCHAR2(30),

DATA DATE; toate comenziile efectuate asupra schemei.

```
CREATE TABLE COMENZI
```

```
    (UTILIZATOR VARCHAR2(30),
```

```
    NUME_BD VARCHAR2(50),
```

```
    EVENIMENT VARCHAR2(20),
```

```
    NUME_OBIECT VARCHAR2(30),
```

```
    DATA DATE);
```

```
CREATE OR REPLACE TRIGGER inregistrare_comenzi
```

```
    AFTER CREATE OR DROP OR ALTER ON SCHEMA
```

```
BEGIN
```

```
    INSERT INTO COMENZI
```

```
        VALUES (SYS.LOGIN_USER, SYS.DATABASE_NAME, SYS.SYSEVENT,
```

```
        SYS.DICTIONARY_OBJ_NAME, SYSDATE);
```

```
END;
```

```
/
```

```

CREATE TABLE COMENZI
    (UTILIZATOR VARCHAR2(30),
    NUME_BD VARCHAR2(50),
    EVENIMENT VARCHAR2(20),
    NUME_OBIECT VARCHAR2(30),
    DATA DATE);
CREATE OR REPLACE TRIGGER inregistrare_comenzi
    AFTER CREATE OR DROP OR ALTER ON SCHEMA
BEGIN
    INSERT INTO COMENZI
        VALUES (SYS.LOGIN_USER, SYS.DATABASE_NAME, SYS.SYSEVENT, SYS.DICTIONARY_OBJ_NAME, SYSDATE);
END;
/

```

	UTILIZATOR	NUME_BD	EVENIMENT	NUME_OBIECT	DATA
1	PROIECT	ORCLPDB1	CREATE	ATRIBUIRE_TASKURI	12-JAN-24
2	PROIECT	ORCLPDB1	CREATE	AFISARE_TASKURI	12-JAN-24
3	PROIECT	ORCLPDB1	CREATE	AFISARE_TASKURI	12-JAN-24
4	PROIECT	ORCLPDB1	CREATE	ACTUALIZARE_BUGET	12-JAN-24
5	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	12-JAN-24
6	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	12-JAN-24
7	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	12-JAN-24
8	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	12-JAN-24
9	PROIECT	ORCLPDB1	CREATE	MODIFICARI_ALUMNI	12-JAN-24
10	PROIECT	ORCLPDB1	CREATE	MODIFICARI_ALUMNI	12-JAN-24
11	PROIECT	ORCLPDB1	CREATE	ACTUALIZARE_ALUMNI	12-JAN-24
12	PROIECT	ORCLPDB1	ALTER	PROIECT	10-JAN-24
13	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	10-JAN-24
14	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	10-JAN-24
15	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	10-JAN-24
16	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	10-JAN-24
17	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	10-JAN-24
18	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	10-JAN-24
19	PROIECT	ORCLPDB1	CREATE	CEL_MAI_ACTIV_MEMBRU	10-JAN-24
20	PROIECT	ORCLPDB1	CREATE	PACHET	11-JAN-24

13. Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.

```

CREATE OR REPLACE PACKAGE pachet AS
    PROCEDURE atribuire_taskuri;
    PROCEDURE afisare_taskuri;
    FUNCTION actualizare_buget
        (p_id_proiect PROIECT.ID_PROIECT%TYPE)
        RETURN SPONSORI.SUMA%TYPE;
    PROCEDURE cel_mai_activ_membru;
END pachet;
/
CREATE OR REPLACE PACKAGE BODY pachet AS
    PROCEDURE atribuire_taskuri
    IS
        statusul TASK.STATUS%TYPE;

    TYPE TASK_TABLE IS TABLE OF TASK%ROWTYPE INDEX BY PLS_INTEGER;
    TYPE TASK_VARRAY IS VARRAY(100) OF TASK%ROWTYPE;
    TYPE TASK_MEMBRI IS TABLE OF TASK.ID_MEMBRU%TYPE;

        contor NUMBER := 1;
        taskuri_realizate TASK_TABLE;
        taskuri_fara_membru TASK_VARRAY := TASK_VARRAY();
        id_membri_asignati TASK_MEMBRI := TASK_MEMBRI();

    CURSOR c_taskuri_realizate IS
        SELECT *
        FROM TASK
        WHERE STATUS = 'Realizat';

    CURSOR c_taskuri_fara_membru IS
        SELECT *
        FROM TASK
        WHERE ID_MEMBRU IS NULL;

    CURSOR c_taskuri_cu_membru IS
        SELECT DISTINCT ID_MEMBRU

```

```

FROM TASK
WHERE ID_MEMBRU IS NOT NULL;

CURSOR c_membri_activi IS
  SELECT ID_MEMBRU
    FROM MEMBRI_ACTIVI;

BEGIN
  FOR task_rec IN c_taskuri_realizate LOOP
    taskuri_realizate(taskuri_realizate.COUNT + 1) := task_rec;
  END LOOP;

  FOR i IN 1..taskuri_realizate.COUNT LOOP
    DELETE
      FROM TASK
      WHERE ID_TASK = taskuri_realizate(i).ID_TASK;
  END LOOP;

  FOR task_rec IN c_taskuri_fara_membru LOOP
    taskuri_fara_membru.EXTEND;
    taskuri_fara_membru(taskuri_fara_membru.LAST) := task_rec;
  END LOOP;

  FOR task_rec IN c_taskuri_cu_membru LOOP
    id_membri_asignati.EXTEND;
    id_membri_asignati(id_membri_asignati.LAST) := task_rec.ID_MEMBRU;
  END LOOP;

  FOR membru_rec IN c_membri_activi LOOP
    IF contor <= taskuri_fara_membru.COUNT THEN
      UPDATE TASK
        SET ID_MEMBRU = membru_rec.ID_MEMBRU
        WHERE ID_TASK = taskuri_fara_membru(contor).ID_TASK;
      contor := contor + 1;
    ELSE
      EXIT;
    END IF;
  END LOOP;

```

```

END LOOP;
END atribuire_taskuri;

PROCEDURE afisare_taskuri
IS
    v_id_membru TASK.ID_MEMBRU%TYPE;

    CURSOR c_membri_activi IS
        SELECT ID_MEMBRU
        FROM MEMBRI_ACTIVI;

    CURSOR c_taskuri_membri(p_id_membru TASK.ID_MEMBRU%TYPE) IS
        SELECT *
        FROM TASK
        WHERE ID_MEMBRU = p_id_membru;

BEGIN
    FOR membri_rec IN c_membri_activi LOOP
        v_id_membru := membri_rec.ID_MEMBRU;
        DBMS_OUTPUT.PUT_LINE('ID_MEMBRU: ' || v_id_membru);
        FOR task_rec IN c_taskuri_membri(v_id_membru) LOOP
            DBMS_OUTPUT.PUT_LINE(' TASK ID: ' || task_rec.ID_TASK || ', Status: ' ||
task_rec.STATUS);
        END LOOP;
        DBMS_OUTPUT.PUT_LINE('-----');
    END LOOP;
END afisare_taskuri;

FUNCTION actualizare_buget
    (p_id_proiect PROIECT.ID_PROIECT%TYPE)
RETURN SPONSORI.SUMA%TYPE IS
    buget SPONSORI.SUMA%TYPE;
    v_titlu_proiect PROIECT.TITLU_PROIECT%TYPE;
    v_nume_club CLUBURI.NUME_CLUB%TYPE;
    PROIECT_INTERN EXCEPTION;
    BUGET_INSUFICIENT EXCEPTION;

```

```

BEGIN
    SELECT
        SUM(S.SUMA) - 1500, P.TITLU_PROIECT, C.NUME_CLUB
    INTO buget, v_titlu_proiect, v_nume_club
    FROM SPONSORI S
    JOIN PROIECT P
    ON(S.ID_PROIECT = P.ID_PROIECT)
    JOIN CLUBURI C
    ON(P.ID_CLUB = C.ID_CLUB)
    WHERE S.ID_PROIECT = p_id_proiect
    GROUP BY P.TITLU_PROIECT, C.NUME_CLUB;

    IF buget + 1500 <= 300 THEN
        RAISE PROIECT_INTERN;
    END IF;

    IF buget < 0 THEN
        RAISE BUGET_INSUFICIENT;
    END IF;

    DBMS_OUTPUT.PUT_LINE('Proiectul ' || v_titlu_proiect || ' din clubul ' ||
    v_nume_club || ' va ramane cu un buget de ');
    RETURN buget;

EXCEPTION
    WHEN NO_DATA_FOUND OR PROIECT_INTERN THEN
        RAISE_APPLICATION_ERROR(-20004, 'Proiectul este intern si nu are nevoie de
materiale!');

    WHEN BUGET_INSUFICIENT THEN
        RAISE_APPLICATION_ERROR(-20003, 'Bugetul este prea mic pentru realizarea
cumparaturilor!');

END actualizare_buget;

PROCEDURE cel_mai_activ_membru

```

```

IS
  v_id_membru MEMBRI_ACTIVI.ID_MEMBRU%TYPE;
  v_punctaj NUMBER;
BEGIN
  SELECT ID_MEMBRU, punctaj INTO v_id_membru, v_punctaj
  FROM (
    SELECT M.ID_MEMBRU,
      COUNT(T.ID_MEMBRU) + 2 * COUNT(P.ID_COORDONATOR) AS punctaj,
      RANK() OVER (ORDER BY COUNT(T.ID_MEMBRU) + 2 *
      COUNT(P.ID_COORDONATOR) DESC) AS rnk
    FROM CLUBURI C
    JOIN DEPARTAMENTE D ON (C.ID_CLUB = D.ID_CLUB)
    JOIN MEMBRI_ACTIVI M ON (D.ID_DEPARTAMENT = M.ID_DEPARTAMENT)
    LEFT JOIN TASK T ON (M.ID_MEMBRU = T.ID_MEMBRU)
    LEFT JOIN PROIECT P ON (M.ID_MEMBRU = P.ID_COORDONATOR)
    GROUP BY M.ID_MEMBRU
  ) WHERE rnk = 1;

  DBMS_OUTPUT.PUT_LINE('ID Membru cu punctaj maxim: ' || v_id_membru || ','
  Punctaj: ' || v_punctaj);

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20005, 'Un departament nu are membri.');
  WHEN TOO_MANY_ROWS THEN
    RAISE_APPLICATION_ERROR(-20006, 'Un departament are mai multi membri
    cu acelasi punctaj.');
END cel_mai_activ_membru;
END pachet;
/

```

```
1 | CREATE OR REPLACE PACKAGE pachet AS
2 |     PROCEDURE atribuire_taskuri;
3 |     PROCEDURE afisare_taskuri;
4 |     FUNCTION actualizare_buget
5 |         (p_id_proiect PROIECT.ID_PROIECT%TYPE,
6 |          p_id_club CLUBURI.ID_CLUB%TYPE)
7 |         RETURN SPONSORI.SUMA%TYPE;
8 |     PROCEDURE cel_mai_activ_membru;
9 | END pachet;
10 |
11 |CREATE OR REPLACE PACKAGE BODY pachet AS
12 |    PROCEDURE atribuire_taskuri
13 |    IS
14 |        statusul TASK.STATUS%TYPE;
15 |
16 |        TYPE TASK_TABLE IS TABLE OF TASK%ROWTYPE INDEX BY PLS_INTEGER;
17 |        TYPE TASK_VARRAY IS VARRAY(100) OF TASK%ROWTYPE;
18 |        TYPE TASK_MEMBRI IS TABLE OF TASK.ID_MEMBRU%TYPE;
19 |
20 |        cursor NUMBERD -- 1.
```

Script Output x

Task completed in 0,035 seconds

Package PACHET compiled

Package Body PACHET compiled