Elias Posluk
h16elipo
GIK23M

Laboration 5

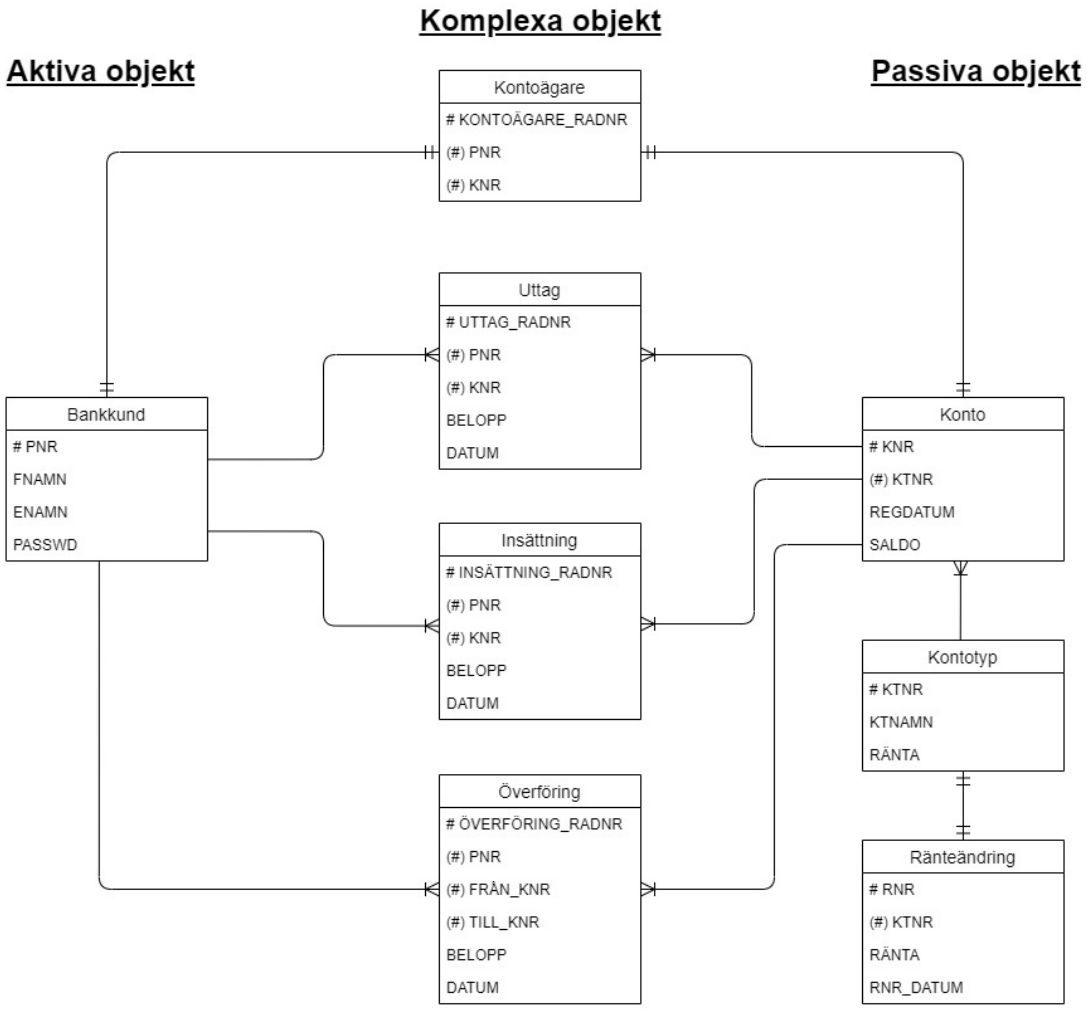
Uppgift 1

```
---------uppgift 5
-----------------------------------bankkund
create table bankkund(
pnr varchar2(11) not null,
fnamn varchar2(25) not null,
enamn varchar2(25) not null,
passwd varchar2(16) not null);

alter table bankkund
add constraint bkund_pnr_pk primary key(pnr)
add constraint bkund_passwd_uq unique(passwd);
-----------------------------------kontotyp
create table kontotyp(
ktnr number(6) not null,
ktnamn varchar2(20) not null,
ränta number(5,2) not null);

alter table kontotyp
add constraint kontotyp_ktnr_pk primary key(ktnr);
-----------------------------------ränteändring
create table ränteändring(
rnr number(6) not null,
ktnr number(6) not null,
ränta number(5,2) not null,
rnr_datum date not null);

alter table ränteändring
add constraint ränteändring_rnr_pk primary key(rnr)
add constraint ränteändring_ktnr_fk foreign key(ktnr) references
kontotyp(ktnr);
-----------------------------------konto
create table konto(
knr number(8) not null,
ktnr number(6) not null,
regdatum date not null,
saldo number(10,2));

alter table konto
add constraint konto_knr_pk primary key(knr)
add constraint konto_ktnr_fk foreign key(ktnr) references
kontotyp(ktnr);
```

```
------------------------------------------------- kontoägare
create table kontoägare(
kontoägare_radnr number(9) not null,
pnr varchar2(11) not null,
knr number(8) not null);

alter table kontoägare
add constraint kontoägare_radnr_pk primary key(kontoägare_radnr)
add constraint kontoägare_pnr_fk foreign key(pnr) references
bankkund(pnr)
add constraint kontoägare_knr_fk foreign key(knr) references
konto(knr);
------------------------------------------------- uttag
create table uttag(
uttag_radnr number(9) not null,
pnr varchar2(11) not null,
knr number(8) not null,
belopp number(10,2),
datum date not null);

alter table uttag
add constraint uttag_radnr_pk primary key(uttag_radnr)
add constraint uttag_pnr_fk foreign key(pnr) references bankkund(pnr)
add constraint uttag_knr_fk foreign key(knr) references konto(knr);
------------------------------------------------- insättning
create table insättning(
insättning_radnr number(9) not null,
pnr varchar2(11) not null,
knr number(8) not null,
belopp number(10,2),
datum date not null);

alter table insättning
add constraint insättning_radnr_pk primary key(insättning_radnr)
add constraint insättning_pnr_fk foreign key(pnr) references
bankkund(pnr)
add constraint insättning_knr_fk foreign key(knr) references
konto(knr);
------------------------------------------------- överföring
create table överföring(
överföring_radnr number(9) not null,
pnr varchar2(11) not null,
från_knr number(8) not null,
till_knr number(8) not null,
belopp number(10,2),
datum date not null);

alter table överföring
add constraint överföring_radnr_pk primary key(överföring_radnr)
add constraint överföring_pnr_fk foreign key(pnr) references
bankkund(pnr)
add constraint överföring_från_knr_fk foreign key(från_knr) references
konto(knr)
add constraint överföring_till_knr_fk foreign key(till_knr) references
konto(knr);
```

```
--Insert Data
--Insert into Kontotyp
INSERT INTO kontotyp(ktnr,ktnamn,ränta)
VALUES(1,'bondkonto',3.4);
INSERT INTO kontotyp(ktnr,ktnamn,ränta)
VALUES(2,'potatiskonto',4.4);
INSERT INTO kontotyp(ktnr,ktnamn,ränta)
VALUES(3,'griskonto',2.4);
COMMIT;

--Insert into Konto
INSERT INTO konto(knr,ktnr,regdatum,saldo)
VALUES(123,1,SYSDATE - 321,0);
INSERT INTO konto(knr,ktnr,regdatum,saldo)
VALUES(5899,2,SYSDATE - 2546,0);
INSERT INTO konto(knr,ktnr,regdatum,saldo)
VALUES(5587,3,SYSDATE - 10,0);
INSERT INTO konto(knr,ktnr,regdatum,saldo)
VALUES(8896,1,SYSDATE - 45,0);
COMMIT;

--Insert into kontoägare
INSERT INTO kontoägare(radnr,pnr,knr)
VALUES(radnr_seq.NEXTVAL,'540126-1111',123);
INSERT INTO kontoägare(radnr,pnr,knr)
VALUES(radnr_seq.NEXTVAL,'691124-4478',123);
INSERT INTO kontoägare(radnr,pnr,knr)
VALUES(radnr_seq.NEXTVAL,'540126-1111',5899);
INSERT INTO kontoägare(radnr,pnr,knr)
VALUES(radnr_seq.NEXTVAL,'691124-4478',8896);
COMMIT;
--Insert end
```

Uppgift 2



Uppgift 3

Skapa en trigger med namnet biufer_bankkund

```
create or replace trigger biufer_bankkund
before insert or update of passwd
on bankkund
for each row
when(length(new.passwd) != 6)
begin
 raise_application_error(-20001,'Lösenordet får bara vara 6 tecken
lång!');
end;
```

Uppgift 4

Skapa en procedur med namnet do_bankkund.

```
create or replace procedure do_bankkund(
p_pnr in bankkund.pnr%type,
p_fnamn in bankkund.fnamn%type,
p_enamn in bankkund.enamn%type,
p_passwd in bankkund.passwd%type)
as

begin

    insert into bankkund(pnr, fnamn, enamn, passwd)
                values(p_pnr, p_fnamn, p_enamn, p_passwd);
commit;

end;

BEGIN
do_bankkund('540126-1111','Hans','Rosendahl','olle45');
do_bankkund('560126-1111','Hans','Rosengårdh','olle85');
do_bankkund('540126-1457','Lina','Karlsson','asdfgh');
do_bankkund('691124-4478','Leena','Kvist','qwerty');
COMMIT;
END;
/
```

Uppgift 5

Triggertest = `EXEC do_bankkund('691124-4478','Leena','Kvist','qwe');`

```
97  for each row
98  when(length(new.passwd) != 6)
99  begin
100     raise_application_error(-20001,'Lösenordet får bara vara 6 tecken lång!');
101 end;
102
103 ------------------------------------------------------------------------ procedure do_bankkund
104 create or replace procedure do_bankkund(
105 p_pnr in bankkund.pnr%type,
106 p_fnamn in bankkund.fnamn%type,
107 p_enamn in bankkund.enamn%type,
108 p_passwd in bankkund.passwd%type)
109 as
110 begin
111     insert into bankkund(pnr, fnamn, enamn, passwd)
112                 values(p_pnr, p_fnamn, p_enamn, p_passwd);
113 commit;
114
115 end;
116
117 ------------------------------------------------trigger test
118 EXEC do_bankkund('691124-4478','Leena','Kvist','qwe');
```

```
ORA-20001: Lösenordet får bara vara 6 tecken lång! ORA-06512: at "SQL_EYGJUKXJFNFUUZSDGAVVUCBOQ.BIUFER_BANKKUND", line 2
ORA-06512: at "SQL_EYGJUKXJFNFUUZSDGAVVUCBOQ.DO_BANKKUND", line 8
ORA-06512: at line 1
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

Uppgift 6

Börja med att skapa en **sekvens** med namnet radnr_seq.

```
create sequence radnr_seq start with 1 increment by 1;
```

SQL Worksheet

```
156 --------------------------------------------------------------------------
157 create sequence radnr_seq start with 1 increment by 1;
158 ------------------------------------------------------------------------ sequence radnr_seq
159
160 INSERT INTO kontotyp(ktnr,ktnamn,ränta)
161 VALUES(1,'bondkonto',3.4);
162 INSERT INTO kontotyp(ktnr,ktnamn,ränta)
163 VALUES(2,'potatiskonto',4.4);
164 INSERT INTO kontotyp(ktnr,ktnamn,ränta)
165 VALUES(3,'griskonto',2.4);
166 COMMIT;
167 INSERT INTO konto(knr,ktnr,regdatum,saldo)
168 VALUES(123,1,SYSDATE - 321,0);
169 INSERT INTO konto(knr,ktnr,regdatum,saldo)
170 VALUES(5899,2,SYSDATE - 2546,0);
171 INSERT INTO konto(knr,ktnr,regdatum,saldo)
172 VALUES(5587,3,SYSDATE - 10,0);
173 INSERT INTO konto(knr,ktnr,regdatum,saldo)
174 VALUES(8896,1,SYSDATE - 45,0);
175 COMMIT;
176
177 INSERT INTO kontoägare(kontoägare_radnr,pnr,knr)
178 VALUES(radnr_seq.nextval,'540126-1111',123);
179 INSERT INTO kontoägare(kontoägare_radnr,pnr,knr)
180 VALUES(radnr_seq.nextval,'691124-4478',123);
181 INSERT INTO kontoägare(kontoägare_radnr,pnr,knr)
182 VALUES(radnr_seq.nextval,'540126-1111',5899);
183 INSERT INTO kontoägare(kontoägare_radnr,pnr,knr)
184 VALUES(radnr_seq.nextval,'691124-4478',8896);
185 COMMIT;
186 --------------------------------------------------------------------------
187 select *
188 from kontoägare;
189
```

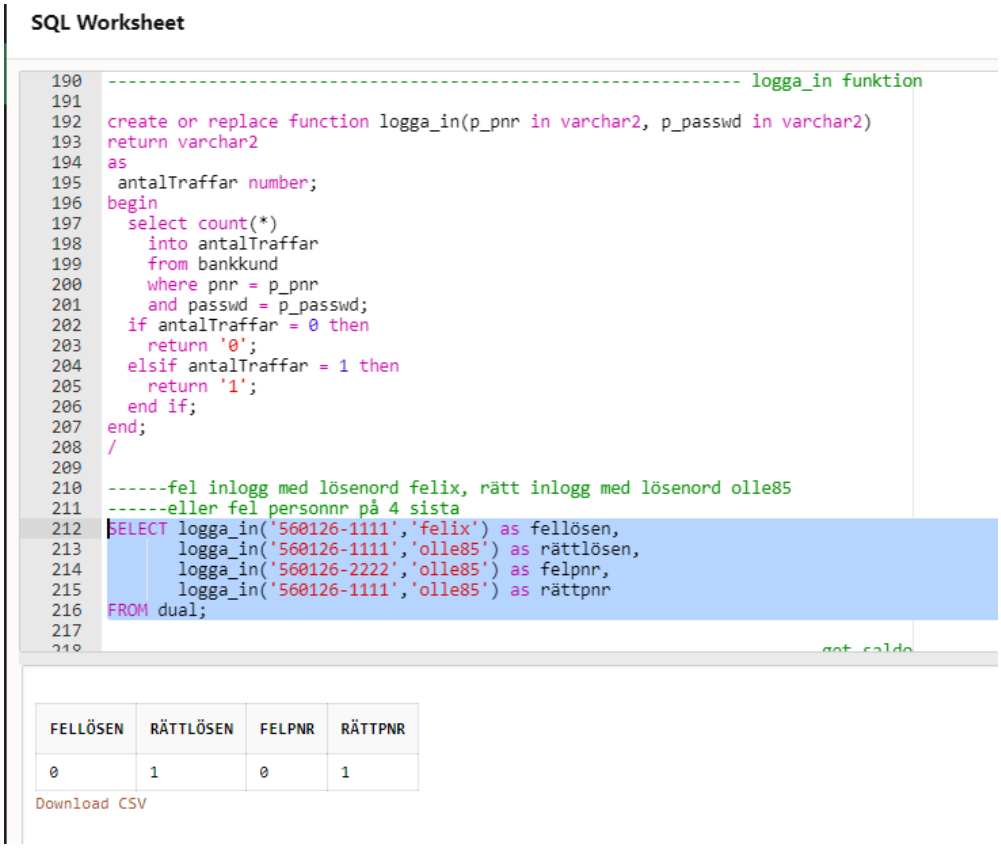
KONTOÄGARE_RADNR	PNR	KNR
1	540126-1111	123
2	691124-4478	123
3	540126-1111	5899
4	691124-4478	8896

Download CSV

Uppgift 7

Skapa en funktion med namnet logga_in. Denna skall returnera 1 eller 0 beroende av hur det gick. Inloggning sker med att en bankund lämnar sitt personnummer och lösenord, dvs kolumnerna pnr och passwd i tabellen bankkund. **Testa att funktionen fungerar** genom att göra:

```
create or replace function logga_in(p_pnr in varchar2, p_passwd in
varchar2)
return varchar2
as
 antalTraffar number;
begin
  select count(*)
    into antalTraffar
    from bankkund
    where pnr = p_pnr
    and passwd = p_passwd;
  if antalTraffar = 0 then
    return '0';
  elsif antalTraffar = 1 then
    return '1';
  end if;
end;
/
```



Uppgift 8

Skapa en funktion med namnet get_saldo.

```
create or replace function get_saldo(konto_id konto.knr%type)
return number
as
  konto_finns number := 0;
  nuvarande_balans konto.knr%type;
begin
  nuvarande_balans := 0;
  select count(*)
  into konto_finns
  from konto
  where knr = konto_id;
  if konto_finns > 0
  then
      select saldo
      into nuvarande_balans
      from konto
      where knr = konto_id;
      return nuvarande_balans;
  else
      konto_finns := -1;
      return konto_finns;
  end if;
end;
```

SQL Worksheet

```
216  FROM dual;
217  ------------------------------------------------------------------------ get_saldo
218  create or replace function get_saldo(konto_id konto.knr%type)
219  return number
220  as
221    konto_finns number := 0;
222    nuvarande_balans konto.knr%type;
223  begin
224    nuvarande_balans := 0;
225
226    select count(*)
227    into konto_finns
228    from konto
229    where knr = konto_id;
230
231    if konto_finns > 0
232    then
233        select saldo
234        into nuvarande_balans
235        from konto
236        where knr = konto_id;
237
238        return nuvarande_balans;
239    else
240        konto_finns := -1;
241        return konto_finns;
242    end if;
243  end;
244
245  select get_saldo(99) ej_konto,
246         get_saldo(5899) Rosendahl,
247         get_saldo(5587) ingen_namn,
248         get_saldo(123) Kvist,
249         get_saldo(8896) ingen_namn
250  from dual;
```

EJ_KONTO	ROSENDAHL	INGEN_NAMN	KVIST	INGEN_NAMN
-1	1000	600	500	200

Download CSV

Uppgift 9

Skapa en funktion med namnet get_behörighet.

```
create or replace function get_behörighet(
p_pnr kontoägare.pnr%type,
p_knr kontoägare.knr%type)
return number
as
konto_id   kontoägare.knr%type;
begin
    select max(kä.knr)
    into konto_id
    from kontoägare kä
    where kä.pnr = p_pnr
    and kä.knr  = p_knr;

           return case when konto_id is not null then 1
              else 0
           end;
end;
```

SQL Worksheet

```
286
287  ---------------------------------------------- get_behörighet
288  create or replace function get_behörighet(
289  p_pnr kontoägare.pnr%type,
290  p_knr kontoägare.knr%type)
291  return number
292  as
293  konto_id   kontoägare.knr%type;
294  begin
295      select max(kä.knr)
296      into konto_id
297      from kontoägare kä
298      where kä.pnr = p_pnr
299      and kä.knr  = p_knr;
300
301          return case when konto_id is not null then 1
302                 else 0
303          end;
304  end;
305
306  select get_behörighet('691124-4478', 123) res_1,
307          get_behörighet('540126-1111', 123) res_3,
308          get_behörighet('650707-1111', 5899) res_2,
309            get_behörighet('861124-4478',5899) res_4
310          from dual;
```

RES_1	RES_3	RES_2	RES_4
1	1	0	0

Download CSV

Uppgift 10
Skapa en trigger med namnet aifer_insättning.

```
create or replace trigger aifer_insättning
after insert
   on insättning
   for each row

declare
    gammal_saldo number(10,2);
    ny_saldo number(10,2);
begin
    select saldo into gammal_saldo from konto where knr = :NEW.knr;

    update konto set saldo = saldo + :NEW.belopp where knr = :NEW.knr;

    select saldo into ny_saldo from konto where knr = :NEW.knr;

    if ny_saldo - gammal_saldo != :NEW.belopp
    then
      raise_application_error(-20001,'Det angivna beloppet är mer än
tillgängligt saldo');
    end if;
end;
```

Uppgift 11
Skapa en trigger med namnet bifer_uttag.

```
create or replace trigger bifer_uttag
before insert
   on uttag
   for each row
declare
    tillgänglig_saldo number(10,2);
begin
    select get_saldo(:NEW.knr) into tillgänglig_saldo from dual;

    if tillgänglig_saldo > -1
    then
      if :NEW.belopp > tillgänglig_saldo
      then
        raise_application_error(-20001,'Du kan inte ta ut ett belopp
mer än det tillgängliga saldot');
      end if;
    else
       raise_application_error(-20001,'Det angivna konto-ID är inte
tillgängligt i systemet');
    end if;
end;
```

Uppgift 12
Skapa en trigger med namnet aifer_uttag.

```
create or replace trigger aifer_uttag
after insert
   on uttag
   for each row
declare
   tillgänglig_saldo number(10,2);
begin
  select saldo into tillgänglig_saldo from konto where knr = :new.knr;

   if tillgänglig_saldo >= :new.belopp
   then
     update konto
     set saldo = saldo - :NEW.belopp
     where knr = :new.knr;
   else
     raise_application_error(-20001,'Det angivna beloppet är mer än
tillgängligt saldo');
   end if;
end;
```

Uppgift 13
Skapa ytterligare en trigger. Denna gång med namnet bifer_överföring.

```
create or replace trigger bifer_överföring
before insert
   on överföring
   for each row
declare
    tillgänglig_saldo number(10,2);
begin
    select get_saldo(:NEW.från_knr) into tillgänglig_saldo from dual;

    if tillgänglig_saldo > -1
    then
      if :NEW.belopp > tillgänglig_saldo
      then
        raise_application_error(-20001,'Du kan inte ta ut mer pengar
än det finns tillgängligt på kontot som du flyttar pengar från');
      end if;
    else
       raise_application_error(-20001,'Det angivna konto-ID är inte
tillgängligt i systemet');
    end if;
end;
```

Uppgift 14

Skapa nu den sista triggern i denna labb. Triggern skall ha namnet aifer_överföring.

```
create or replace trigger aifer_överföring
after insert
   on överföring
   for each row
declare
   tillgänglig_saldo number(10,2);
begin
  select saldo into tillgänglig_saldo from konto where knr =
:NEW.från_knr;

  if tillgänglig_saldo >= :NEW.belopp
  then
    update konto set saldo = saldo - :NEW.belopp
    where knr = :NEW.från_knr;

    update konto set saldo = saldo + :NEW.belopp
    where knr = :NEW.till_knr;
  else
     raise_application_error(-20001,'Det angivna beloppet är mer än
tillgängligt saldo');
  end if;
end;
```

Uppgift 15

Skapa en procedur med namnet do_insättning.

```
create or replace procedure do_insättning
as
  ny_id insättning.insättning_radnr%type;
  kund_id bankkund.pnr%type;
  konto_id konto.knr%type;
  insättning_belopp insättning.belopp%type;
  insättning_datum insättning.datum%type;
  ny_saldo konto.saldo%type;

begin
  kund_id := '691124-4478'; konto_id := 123; insättning_belopp :=
100000; insättning_datum := SYSDATE;

  select nvl((max(insättning_radnr)+1),1) into ny_id from insättning;

  insert into insättning (insättning_radnr, pnr, knr, belopp,datum)
  values(ny_id, kund_id, konto_id, insättning_belopp,
insättning_datum);

  select saldo into ny_saldo from konto where knr = konto_id;

  dbms_output.put_line('Nya beloppet för konto-id ' || konto_id || '
är: ' || ny_saldo);
end;
```

Uppgift 16

Testa att proceduren do_insättning fungerar.

```
445   ny_id insättning.insättning_radnr%type;
446   kund_id bankkund.pnr%type;
447   konto_id konto.knr%type;
448   insättning_belopp insättning.belopp%type;
449   insättning_datum insättning.datum%type;
450   ny_saldo konto.saldo%type;
451
452 begin
453   kund_id := '691124-4478'; konto_id := 123; insättning_belopp := 100000; insättning_datum := SYSDATE;
454
455   select nvl((max(insättning_radnr)+1),1) into ny_id from insättning;
456
457   insert into insättning (insättning_radnr, pnr, knr, belopp,datum)
458   values(ny_id, kund_id, konto_id, insättning_belopp, insättning_datum);
459
460   select saldo INTO ny_saldo from konto where knr = konto_id;
461
462   dbms_output.put_line('Nya beloppet för konto-id ' || konto_id || ' är: ' || ny_saldo);
463 end;
464
465 select *
466 from insättning;
467
468 select *
469 from konto;
470
471 select
472
473 select *
```

KNR	KTNR	REGDATUM	SALDO
123	1	25-JUL-19	0
5899	2	21-JUN-13	0
5587	3	31-MAY-20	0
8896	1	26-APR-20	0

Download CSV
4 rows selected.

Figur 1 Innan insättning, inga pengar i konton

SQL Worksheet

```
450   ny_saldo konto.saldo%type;
451
452 begin
453   kund_id := '691124-4478'; konto_id := 123; insättning_belopp := 100000; insättning_datum := SYSDATE;
454
455   select nvl((max(insättning_radnr)+1),1) into ny_id from insättning;
456
457   insert into insättning (insättning_radnr, pnr, knr, belopp,datum)
458   values(ny_id, kund_id, konto_id, insättning_belopp, insättning_datum);
459
460   select saldo INTO ny_saldo from konto where knr = konto_id;
461
462   dbms_output.put_line('Nya beloppet för konto-id ' || konto_id || ' är: ' || ny_saldo);
463 end;
464
465 select *
466 from insättning;
467
468 select *
469 from konto;
470
471 select
472
473 select *
474 from bankkund;
475
476 begin
477 do_insättning;
478 end;
```

```
Statement processed.
Nya beloppet för konto-id 123 är: 100000
```

Figur 2 inför insättning

```
450   ny_saldo konto.saldo%type;
451
452 begin
453   kund_id := '691124-4478'; konto_id := 123; insättning_belopp := 100000; insättning_datum := SYSDATE;
454
455   select nvl((max(insättning_radnr)+1),1) into ny_id from insättning;
456
457   insert into insättning (insättning_radnr, pnr, knr, belopp,datum)
458   values(ny_id, kund_id, konto_id, insättning_belopp, insättning_datum);
459
460   select saldo INTO ny_saldo from konto where knr = konto_id;
461
462   dbms_output.put_line('Nya beloppet för konto-id ' || konto_id || ' är: ' || ny_saldo);
463 end;
464
465 select *
466 from insättning;
467
468 select *
469 from konto;
470
471 select
472
473 select *
474 from bankkund;
475
476 begin
477 do_insättning;
478 end;
```

KNR	KTNR	REGDATUM	SALDO
123	1	25-JUL-19	100000
5899	2	21-JUN-13	0
5587	3	31-MAY-20	0
8896	1	26-APR-20	0

Download CSV
4 rows selected.

Figur 3 Efter insättning, pengar i kontot till 123

Uppgift 17
Skapa en procedur som heter do_uttag.

```
create or replace procedure do_uttag
as
  verifierad integer;
  ny_id uttag.uttag_radnr%type;
  kund_id bankkund.pnr%type;
  konto_id konto.knr%type;
  uttag_belopp uttag.belopp%type;
  uttag_datum uttag.datum%type;
  ny_saldo konto.saldo%type;

  obehörig exception;

begin
kund_id := '691124-4478'; konto_id := 123; uttag_belopp := 100000;
uttag_datum := SYSDATE;

  select get_behörighet('691124-4478',123) into verifierad from dual;

  if(verifierad = 1)
  then
  select nvl((max(uttag_radnr)+1),1) into ny_id from uttag;

  insert into uttag (uttag_radnr, pnr, knr, belopp,datum)
  values(ny_id, kund_id, konto_id, uttag_belopp, uttag_datum);

  select saldo INTO ny_saldo from konto where knr = 123;
  dbms_output.put_line('Den nya saldot för konto-id ' || konto_id || '
är: ' || ny_saldo);

  else
    raise obehörig;
  end if;

  exception
   when obehörig
   then raise_application_error(-20000, 'Obehörig användare!');

end;
```

Uppgift 18

Testa att proceduren do_uttag fungerar.

```
502   then
503   select nvl((max(uttag_radnr)+1),1) into ny_id from uttag;
504
505   insert into uttag (uttag_radnr, pnr, knr, belopp,datum)
506   values(ny_id, kund_id, konto_id, uttag_belopp, uttag_datum);
507
508   select saldo INTO ny_saldo from konto where knr = 123;
509
510    dbms_output.put_line('Den nya saldot för konto-id ' || konto_id || ' är: ' || ny_saldo);
511
512   else
513     raise obehörig;
514   end if;
515
516   exception
517    when obehörig
518    then raise_application_error(-20000, 'Obehörig användare!');
519
520 end;
521
522 update konto set saldo = 1000;
523
524 begin
525 do_uttag;
526 end;
527
528 select *
529 from konto;
530
```

KNR	KTNR	REGDATUM	SALDO
123	1	25-JUL-19	1000
5899	2	21-JUN-13	1000
5587	3	31-MAY-20	1000
8896	1	26-APR-20	1000

Download CSV
4 rows selected.

Figur 4 innan uttag, alla konton har pengar.

```
  then
  select nvl((max(uttag_radnr)+1),1) into ny_id from uttag;

  insert into uttag (uttag_radnr, pnr, knr, belopp,datum)
  values(ny_id, kund_id, konto_id, uttag_belopp, uttag_datum);

  select saldo INTO ny_saldo from konto where knr = 123;

   dbms_output.put_line('Den nya saldot för konto-id ' || konto_id || ' är: ' || ny_saldo);

  else
    raise obehörig;
  end if;

  exception
   when obehörig
   then raise_application_error(-20000, 'Obehörig användare!');

end;

update konto set saldo = 1000;

begin
do_uttag;
end;

select *
from konto;
```

```
Statement processed.
Den nya saldot för konto-id 123 är: 950
```

Figur 5 tar ut 50kr från kontot 123, kvar finns 950kr.

```
  if(verifierad = 1)
  then
  select nvl((max(uttag_radnr)+1),1) into ny_id from uttag;

  insert into uttag (uttag_radnr, pnr, knr, belopp,datum)
  values(ny_id, kund_id, konto_id, uttag_belopp, uttag_datum);

  select saldo INTO ny_saldo from konto where knr = 123;

   dbms_output.put_line('Den nya saldot för konto-id ' || konto_id || ' är: ' || ny_saldo);

  else
    raise obehörig;
  end if;

  exception
   when obehörig
   then raise_application_error(-20000, 'Obehörig användare!');

end;

update konto set saldo = 1000;

begin
do_uttag;
end;

select *
from konto;
```

```
ORA-20000: Obehörig användare! ORA-06512: at "SQL_EYGJUKXJFNFUUZSDGAVVUCBOQ.DO_UTTAG", line 35
ORA-06512: at line 2
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

Figur 6 försöker ta ut pengar som obehörig användare, inte tillåtet.

```
  then
  select nvl((max(uttag_radnr)+1),1) into ny_id from uttag;

  insert into uttag (uttag_radnr, pnr, knr, belopp,datum)
  values(ny_id, kund_id, konto_id, uttag_belopp, uttag_datum);

  select saldo INTO ny_saldo from konto where knr = 123;

   dbms_output.put_line('Den nya saldot för konto-id ' || konto_id || ' är: ' || ny_saldo);

  else
    raise obehörig;
  end if;

  exception
   when obehörig
   then raise_application_error(-20000, 'Obehörig användare!');

end;

update konto set saldo = 1000;

begin
do_uttag;
end;

select *
from konto;
```

```
ORA-20001: Du kan inte ta ut ett belopp mer än det tillgängliga saldot ORA-06512: at "SQL_EYGJUKXJFNFUUZSDGAVVUCBOQ.BIFER_UTTAG", line 11
ORA-06512: at "SQL_EYGJUKXJFNFUUZSDGAVVUCBOQ.DO_UTTAG", line 22
ORA-06512: at line 2
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

Figur 7 försöker ta ut mer än vad det finns i kontot.

Uppgift 19
Skapa labbens sista objekt som är en procedur med namnet do_överföring.

```
create or replace procedure do_överföring
as
  verifierad integer := 0;
  ny_id överföring.överföring_radnr%type;
  kund_id bankkund.pnr%type;
  avsändare_konto_knr konto.knr%type;
  mottagare_konto_knr konto.knr%type;
  överföring_belopp överföring.belopp%type;
  överföring_datum överföring.datum%type;
  avsändare_saldo konto.saldo%type;
  mottagare_saldo konto.saldo%type;

  obehörig exception;

begin
  kund_id := '691124-4478'; avsändare_konto_knr := 123;
mottagare_konto_knr := 5899; överföring_belopp := 155;
överföring_datum := SYSDATE;

  select get_behörighet(kund_id,avsändare_konto_knr) into verifierad
from dual;

  if(verifierad = 1)
  then
  select nvl((max(överföring_radnr)+1),1) into ny_id from överföring;

  insert into överföring(överföring_radnr, pnr, från_knr, till_knr,
belopp, datum)
  values(ny_id, kund_id, avsändare_konto_knr, mottagare_konto_knr,
överföring_belopp, överföring_datum);

  select saldo into avsändare_saldo from konto where knr =
avsändare_konto_knr;
  select saldo into mottagare_saldo from konto where knr =
mottagare_konto_knr;

  dbms_output.put_line('Den nya saldot för avsändaren med konto-id '
|| avsändare_konto_knr || ' är : '|| avsändare_saldo);
  dbms_output.put_line('Den nya saldot för mottagaren med konto-id '
|| mottagare_konto_knr || ' är : '|| mottagare_saldo);

  else
    raise obehörig;
  end if;

  exception
   when obehörig
   then raise_application_error(-20000, 'Obehörig användare!');
end;
```

Uppgift 20

Testa att flytta pengar mellan konton med proceduren do_överföring. Testa även här att triggrarna fungerar.

SQL Worksheet

```
568   select saldo into mottagare_saldo from konto where knr = mottagare_konto_knr;
569
570   dbms_output.put_line('Den nya saldot för avsändaren med konto-id ' || avsändare_konto_knr || ' är : '|| avsändare_saldo);
571   dbms_output.put_line('Den nya saldot för mottagaren med konto-id ' || mottagare_konto_knr || ' är : '|| mottagare_saldo);
572
573   else
574     raise obehörig;
575   end if;
576
577   exception
578    when obehörig
579    then raise_application_error(-20000, 'Obehörig användare!');
580
581 end;
582
583 update konto set saldo = 1000;
584
585 select *
586 from konto;
587
588 select *
589 from överföring;
590
591 begin
592 do_överföring;
593 end;
594
595 delete from överföring;
596
```

KNR	KTNR	REGDATUM	SALDO
123	1	25-JUL-19	1000
5899	2	21-JUN-13	1000
5587	3	31-MAY-20	1000
8896	1	26-APR-20	1000

Download CSV
4 rows selected.

Figur 8 Alla har samma summa i kontot, innan överföring

SQL Worksheet

```
568   select saldo into mottagare_saldo from konto where knr = mottagare_konto_knr;
569
570   dbms_output.put_line('Den nya saldot för avsändaren med konto-id ' || avsändare_konto_knr || ' är : '|| avsändare_saldo);
571   dbms_output.put_line('Den nya saldot för mottagaren med konto-id ' || mottagare_konto_knr || ' är : '|| mottagare_saldo);
572
573   else
574     raise obehörig;
575   end if;
576
577   exception
578    when obehörig
579    then raise_application_error(-20000, 'Obehörig användare!');
580
581 end;
582
583 update konto set saldo = 1000;
584
585 select *
586 from konto;
587
588 select *
589 from överföring;
590
591 begin
592 do_överföring;
593 end;
594
595 delete from överföring;
596
```

```
Statement processed.
Den nya saldot för avsändaren med konto-id 123 är : 500
Den nya saldot för mottagaren med konto-id 5899 är : 1500
```

Figur 9 Överföring mellan konton

SQL Worksheet

```
568   select saldo into mottagare_saldo from konto where knr = mottagare_konto_knr;
569
570   dbms_output.put_line('Den nya saldot för avsändaren med konto-id ' || avsändare_konto_knr || ' är : '|| avsändare_saldo);
571   dbms_output.put_line('Den nya saldot för mottagaren med konto-id ' || mottagare_konto_knr || ' är : '|| mottagare_saldo);
572
573   else
574     raise obehörig;
575   end if;
576
577   exception
578    when obehörig
579    then raise_application_error(-20000, 'Obehörig användare!');
580
581 end;
582
583 update konto set saldo = 1000;
584
585 select *
586 from konto;
587
588 select *
589 from överföring;
590
591 begin
592 do_överföring;
593 end;
594
595 delete from överföring;
596
```

KNR	KTNR	REGDATUM	SALDO
123	1	25-JUL-19	500
5899	2	21-JUN-13	1500
5587	3	31-MAY-20	1000
8896	1	26-APR-20	1000

Download CSV
4 rows selected.

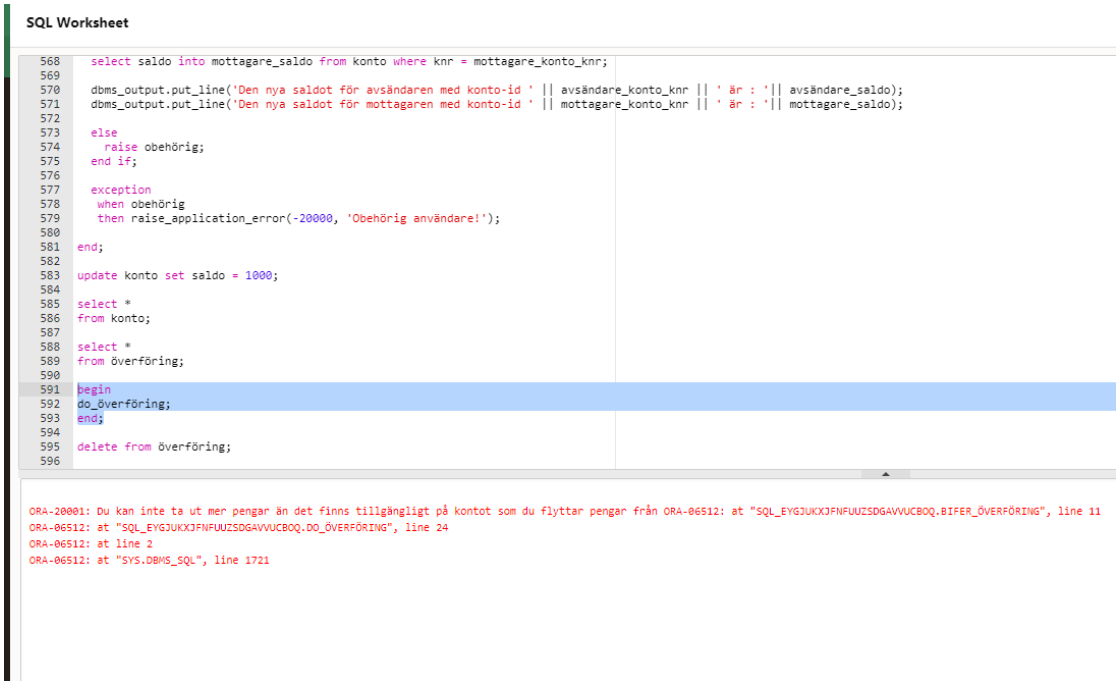
Figur 10 Efter överföring, förändring i respektive konton

SQL Worksheet

```
568   select saldo into mottagare_saldo from konto where knr = mottagare_konto_knr;
569
570   dbms_output.put_line('Den nya saldot för avsändaren med konto-id ' || avsändare_konto_knr || ' är : '|| avsändare_saldo);
571   dbms_output.put_line('Den nya saldot för mottagaren med konto-id ' || mottagare_konto_knr || ' är : '|| mottagare_saldo);
572
573   else
574     raise obehörig;
575   end if;
576
577   exception
578    when obehörig
579    then raise_application_error(-20000, 'Obehörig användare!');
580
581 end;
582
583 update konto set saldo = 1000;
584
585 select *
586 from konto;
587
588 select *
589 from överföring;
590
591 begin
592 do_överföring;
593 end;
594
595 delete from överföring;
596
```

```
ORA-20000: Obehörig användare! ORA-06512: at "SQL_EYGJUKXJFNFUUZSDGAVVUCBOQ.DO_ÖVERFÖRING", line 39
ORA-06512: at line 2
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

Figur 11 om man är obehörig när man försöker överföra pengar



Figur 12 Om man inte har tillräckligt med saldo för att göra överföring