Create a view called active\_consultants that will display the name and job description of the active consultants.

SQL> create view active\_consultants as  
select consultant\_fname, consultant\_sname, job\_desc  
from consultant join job on consultant\_job = job\_id  
where consultant\_status = 'active';

To display view:

SQL> select \* from consultants\_out;

Create a view called consultant\_out which will display the name of the consultant and the number of current contracts they are leading on.  
SQL> create view consultants\_out as  
select consultant\_fname, consultant\_sname, count(\*) as leading\_on  
from consultant join contract on consultant\_id = contract\_lead  
where contract\_end is not null  
group by consultant\_fname, consultant\_sname, consultant\_id;  
  
Create a view called client\_annual\_bill which will display the name and the total charged to a client for the last year.  
SQL> create view client\_annual\_bill as  
select emp\_name, sum((charge\_fee + charge\_expenses)) as total\_due  
from employer join contract on contract\_client = emp\_id  
join charge\_sheet on contract\_id = charge\_job  
where contract\_end > sysdate – 365  
group by emp\_name, emp\_id;  
  
Create a trigger that will automatically calculate the next sequential consultant\_id.  
SQL> create or replace trigger trg\_newconsultant  
before insert on consultant  
for each row  
declare  
v\_id number;  
begin  
 select max(consultant\_id) + 1 into v\_id from consultant;  
 :new.consultant\_id := v\_id;  
end;  
/  
To test this trigger:insert into consultant (consultant\_fname, consultant\_sname, consultant\_id, consultant\_job)   
values ('cid','phillips', 25, 2);  
select consultant\_fname, consultant\_sname, consultant\_id from consultant;  
  
Create 2 triggers on charge\_sheet, one to overwrite date with the next sequential charge id and the other to calculate the expenses for the job being charged for.  
SQL> create or replace trigger trg\_chargeexpenses  
before insert on charge\_sheet  
for each row  
declare  
v\_expenses number;  
begin  
 select sum(expense\_value) into v\_expenses from job\_expenses   
 where expense\_job = :new.charge\_job;  
 :new.charge\_expenses := v\_expenses;  
end;  
/  
SQL> create or replace trigger trg\_newcharge  
before insert on charge\_sheet  
for each row  
declare  
v\_id number;  
begin  
 select max(charge\_id) + 1 into v\_id from charge\_sheet;  
 :new.charge\_id := v\_id;  
end;  
/  
To test these triggers:  
SQL> insert into charge\_sheet values (1, 5, 10000, 78);  
1 row created.  
SQL> select \* from charge\_sheet where charge\_job = 5;  
charge\_id charge\_job charge\_fee charge\_expenses  
---------- ---------- ---------- --------------  
15 5 10000 13940

How much has each employer/client been charged in total to date? Display the name of the client and the total amount charged to each of them.

SQL> select emp\_name,sum(charge\_fee + charge\_expenses)

from employer join contract on emp\_id = contract\_client

join charge\_sheet on charge\_job = contract\_id

group by emp\_name, emp\_id;

Write the trigger that will calculate the next charge\_id.

SQL> create or replace trigger trg\_newcharge

before insert on charge\_sheet

for each row

declare

v\_id number;

begin

select max(charge\_id) + 1 into v\_id

from charge\_sheet;  
 :new.charge\_id := v\_id;

end;

/

Write the trigger that will calculate the fee due.

Work out duration of the contract.

For every day over 10 charge 500.

10000 charge is due as a min, if contract took over 10 days this is added to additional daily rate (if contract took 15 days the fee is 12500)

SQL> create or replace trigger trg\_chargefee

before insert on charge\_sheet

for each row

declare

v\_duration number;

begin

select (contract\_end - contract\_start) into v\_duration

from contract

where contract\_id= :new.charge\_job;

if v\_duration < 11 then

:new.charge\_fee := 10000;

else

:new.charge\_fee := ((v\_duration - 10) \* 500) + 10000;

end if;

end;

/

Write the trigger that will calculate the expenses due.

SQL> create or replace trigger trg\_chargeexpenses

before insert on charge\_sheet

for each row

declare

v\_expenses number;

begin

select sum(expense\_value) into v\_expenses

from job\_expenses where expense\_job = :new.charge\_job

and expense\_approved is not null;

:new.charge\_expenses := v\_expenses;

end;

/

Write the trigger that will insert a new charge sheet when the contract is closed and all expenses are fulfilled.

SQL> create or replace trigger trg\_closecontract

after update of contract\_end on contract

for each row

declare

v\_outstanding number;

begin

select count(\*) into v\_outstanding

from job\_expenses

where expense\_job= :new.contract\_id and expense\_approved is null;

if (v\_outstanding = 0) then

insert into charge\_sheet values (1,:new.contract\_id,10000,999);

end if;

end;

/

Write the trigger that will fire when an expense is approved and if the contract for that expense is closed and all other expenses are approved create a new charge sheet record.

SQL> create or replace trigger trg\_expensecomplete

after update of expense\_approved on job\_expenses

for each row

declare

v\_outstanding number;

v\_end\_date date;

begin

select count(\*) into v\_outstanding

from job\_expenses

where expense\_job= :new.expense\_job and expense\_approved is null;

select contract\_end into v\_end\_date

from contract

where contract\_id = :new.expense\_job;

if v\_outstanding = 0 and v\_end\_date is not null then

insert into charge\_sheet values (1,:new.expense\_job,10000,999);

end if;

end;

/

max(emp\_id)

-----------

14

SQL> insert into employer (emp\_id,emp\_name) values(15,'lilith tinker');

emp\_id emp\_name emp\_addr1

---------- ------------------------------ --------------------------

emp\_addr2 emp\_priority

------------------------------ ------------

15 lilith tinker

3

---------------------------------------------------------------

-- note the emp\_priority was not inserted as a value but the

default from the ddl statment has been used i.e. 3

SQL> select max(contract\_id) from contract;

max(contract\_id)

----------------  
30

SQL> insert into contract values(31,15,25,15,'1-may-19',null,null,'new contract offer');

which consultants are pilots

select consultant\_id, job\_desc

from consultant join job on consultant\_job = job\_id

where job\_desc = 'Pilot';

consultant\_id job\_desc

------------- ---------------

25 pilot

26 pilot

19 pilot

27 pilot

28 pilot

29 pilot

30 pilot

31 pilot

24 pilot

23 pilot

consultant\_id job\_desc

------------- ---------------

22 pilot

21 pilot

20 pilot

SQL> select max(team\_id) from job\_team;

max(team\_id)

------------

60

insert into job\_team values(61,31,26);

SQL> create table cons\_chg\_log

(

consult\_log\_id number(3) constraint conschglog\_pk primary key,

cons\_update\_date date default sysdate,

cons\_update\_comment varchar2(70),

consultant\_id\_log number(3),

consultant\_fname\_old varchar2(20),

consultant\_fname\_new varchar2(20),

consultant\_sname\_old varchar2(20),

consultant\_sname\_new varchar2(20),

consultant\_town\_old varchar2(15),

consultant\_town\_new varchar2(15),

consultant\_job\_old number(3),

consultant\_job\_new number(3),

consultant\_status\_old varchar2(20),

consultant\_status\_new varchar2(20)

);

SQL> create or replace trigger consultant\_change\_log

before update or delete on consultant

for each row

declare

v\_consult\_log\_id number;

begin

select max(consult\_log\_id) + 1 into v\_consult\_log\_id

from cons\_chg\_log;

if v\_consult\_log\_id is null then

v\_consult\_log\_id := 1;

end if;

insert into cons\_chg\_log values(

v\_consult\_log\_id,

sysdate,

user,

:old.consultant\_id,

:old.consultant\_fname,

:new.consultant\_fname,

:old.consultant\_sname,

:new.consultant\_sname,

:old.consultant\_town,

:new.consultant\_town,

:old.consultant\_job,

:new.consultant\_job,

:old.consultant\_status,

:new.consultant\_status

);

end;

/

SQL> desc consultant  
name null? type

----------------------------------------- -------- ----------------------------

consultant\_fname varchar2(20)

consultant\_sname varchar2(20)

consultant\_id not null number(3)

consultant\_town varchar2(15)

consultant\_job number(3)

consultant\_status varchar2(20)

SQL> select max(consultant\_id) from consultant;

max(consultant\_id)

------------------

47

SQL> select \* from job;

job\_id job\_desc

---------- ---------------

1 oracle

2 mercenary

3 army

4 bar staff

5 sales

6 self-employed

7 free lancer

8 undefined

9 warrior

10 thief

11 monk

job\_id job\_desc

---------- ---------------

12 black mage

13 white mage

14 red mage

15 pilot

SQL> insert into consultant values('aleister','crowley',48,'cefalu',12,'active');

1 row created.

SQL> select \* from cons\_chg\_log;

no rows selected

SQL> update consultant

set consultant\_town = 'hebron',consultant\_status = 'inactive/deceased'

where consultant\_id = 48;

delete from consultant where consultant\_id = 48;

set lines 256

set trimout on

set tab off

select \* from cons\_chg\_log;