Année universitaire 2008/2009

Master 1 SIIO
Projet Introduction au Décisionnel, Oracle

Ce projet comporte trois parties sur les thèmes suivants : création de base de donnée, requêtes SQL, mise en œuvre de déclencheurs base de données en PL/SQL. Ces trois parties utilisent le schéma de la base de données de production décrit ci-dessous, extrait du schéma plus complet de la base de données d'une banque. Cette base de données décrit les clients de la banque et les comptes dont ils sont titulaires. Un client peut être titulaire de plusieurs comptes et un compte peut avoir plusieurs titulaires susceptibles d'y appliquer des opérations. Il existe trois types de comptes : les comptes courant, les comptes de revolving et les comptes d'épargne. Pour tout compte, on enregistre son numéro, son type et la balance actuelle du compte (montant disponible pouvant passer en négatif lorsque le compte est à découvert). Pour les comptes courants on enregistre en plus, dans la table *CPTCOURANT*, le montant du découvert autorisé et la date depuis laquelle le compte est à découvert (Null si le compte est positif). On maintient aussi la valeur minimum et la valeur maximum du solde de chaque compte courant, mois par mois, dans la table *CPTCOURANT_MOIS*. Les opérations sur les comptes sont enregistrées dans la table *OPERATION*. Cette table répertorie le compte concerné, le client auteur de l'opération, le type de l'opération, le montant de l'opération, la date de l'opération et l'état de l'opération. Le type de l'opération est soit un débit soit un crédit.

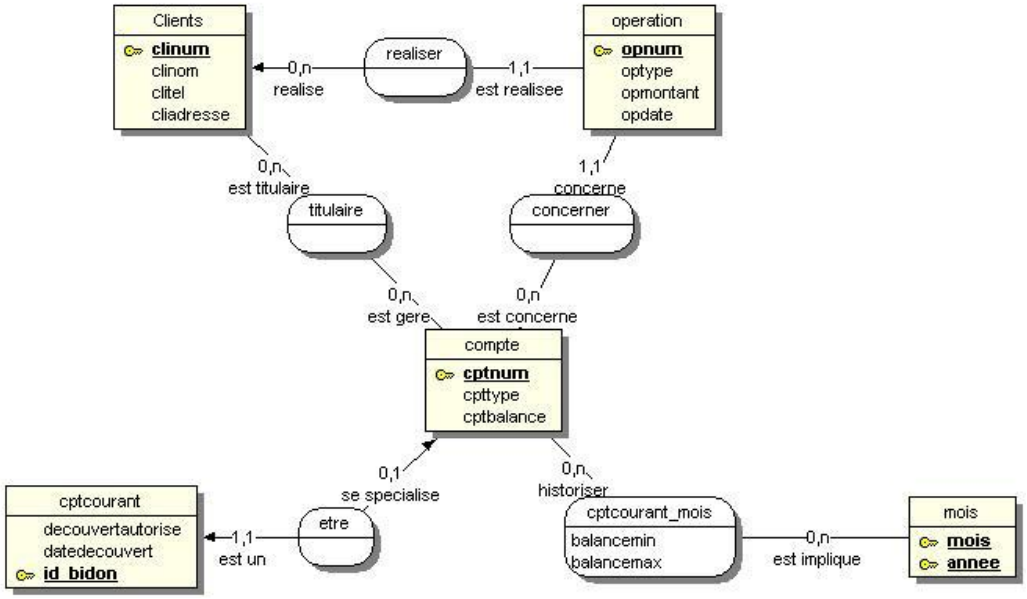
CLIENTS (CliNum, CliNom, CliTel, CliAdresse)
TITULAIRE(CliNum*,CptNum*)
COMPTE(CptNum, CptType, CptBalance)
CPTCOURANT(CptNum*, DecouvertAutorise, DateDecouvert)
CPTCOURANT_MOIS(CptNum*, Mois, Annee, BalanceMin, BalanceMax)
OPERATION(OpNum, CptNum*, CliNum*, OpType, OpMontant, OpDate)

Extrait du dictionnaire données :

CliNum, CliNom	Identifiant et nom d'un client
CliTel, CliAdresse	Téléphone et adresse d'un client
CptNum, CptType	Numéro et type d'un compte (*cpte courant, revolving, épargne*)
CptBalance, CptStatut	Balance d'un compte (montant disponible)
DecouvertAutorise	Montant du découvert autorisé d'un compte
DateDecouvert	Date depuis laquelle le compte est en négatif
Mois, Annee	Mois et année pour lesquels on a calculé les balances min et max
BalanceMin, BalanceMax	Balances minimum et maximum d'un compte au cours d'un mois
OpNum, OpType	Numéro et type d'une opération (*débit, crédit*)
OpMontant, OpDate	Montant et date d'une opération.

Partie I : Créer la base de données

Créer la base de données de production de la banque sur le serveur Oracle SIIP en exploitant le fichier banque.sql placé dans le dossier : *ftp://ftpsip.univ-tlse1.fr/jean-marc.thevenin/M1SIIO/projet/89*.



Insérer 4 clients, 6 comptes et une quinzaine d'opérations de façon à pouvoir tester votre projet.

Pour insérer des lignes dans cette base de données, vous pouvez utiliser la commande SQL suivante : insert into NOMTABLE values (111,'aaa', …) ;

Vous pouvez aussi remplir vos tables en utilisant un client ACCESS qui voit les tables Oracle via un interprète ODBC (commande *fichier/donnée externes/lier les tables* comme nous l'avons fait en TD). Attention, l'interprète doit être recréé chaque fois que vous changez de PC dans les salles d'accès libre.

Enfin, si vous avez respecté les noms de tables et les noms de colonnes donnés dans le schéma ci-dessus, vous pouvez aussi remplir vos tables en exploitant l'application formsBuilder banque.fmb placée dans dossier : *ftp://ftpsip.univ-tlse1.fr/jean-marc.thevenin/M1SIIO/projet/89*.

Partie II : requêtes SQL

1. Afficher le nom et l'adresse de tous les titulaires du compte n° X.
2. Afficher la liste des opérations effectuées sur le compte X au cours du mois de novembre (to_date(OpDate,'mm/yy') = to_date('11/08','mm/yy')).
3. Afficher la valeur du solde minimum au mois de novembre, pour chaque compte dont monsieur 'X' est titulaire.

Remarque : pour tester vos requêtes vous remplacerez les X par des valeurs que vous avez mises dans votre base de données. Attention, Oracle est sensible à la casse (minuscules/majuscules).

Partie III : programmation PL/SQL

Ecrire le programme PL/SQL permettant d'actualiser automatiquement le solde d'un compte lors de l'enregistrement d'une opération.

Ecrire le programme PL/SQL permettant d'actualiser automatiquement la colonne DateDecouvert de la table CPTCOURANT et les colonnes BalanceMin et BalanceMax de la table CPTCOURANT_MOIS lors de la mise à jour du solde d'un compte courant.

Université Toulouse 1 – Science sociale

M1 SIIO
Année 2008 – 2009

Projet :
Introduction au SI décisionnel
Oracle

Etudiants : Le Nguyen PHAM (Groupe 3)
Hamza SMAIL (Groupe 2)

Plan

Introduction.

1. Créer la base de données

- **1.1. Insérer des données pour la table clients**
- **1.2. Insérer des données pour la table compte**
- **1.3. Insérer des données pour la table opération**
- **1.4. Remplir la table Titulaire**

2. Les requêtes SQL

- **2.1. Afficher le nom et l'adresse de tous les titulaires du compte N° X**
- **2.2. Afficher la liste des opérations effectuées sur le compte N° X au cours du mois de novembre**
- **2.3. Afficher la valeur du solde minimum au mois de novembre, pour chaque compte dont Monsieur 'X' est titulaire**

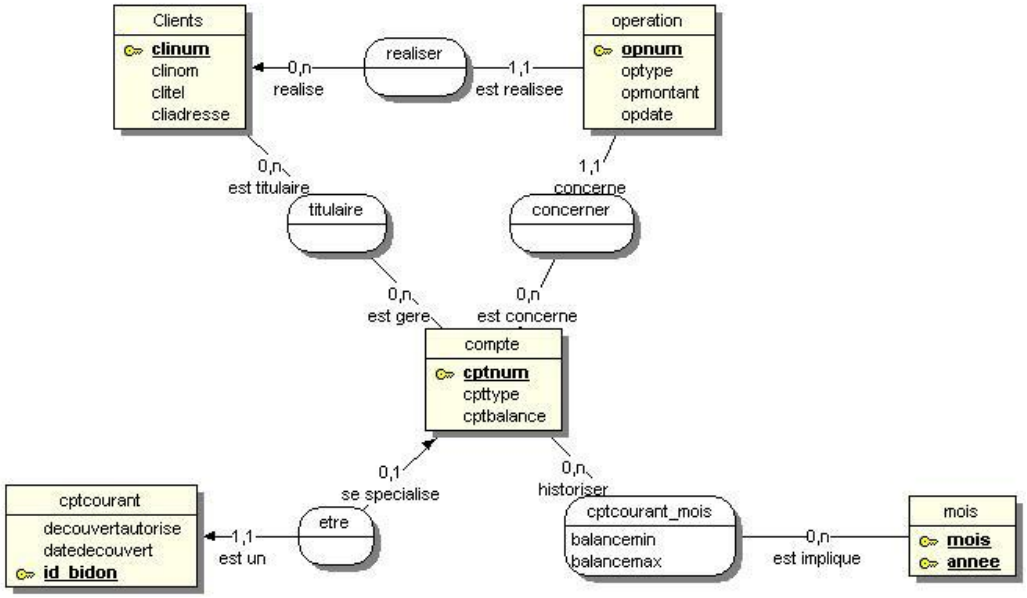
3. Programmation PL/SQL

Conclusion

Introduction

Ce document contient un rapport sur la réalisation de projet dans le cadre de l'Introduction au SI décisionnel (Oracle).

Ce projet comporte trois parties sur les thèmes suivants : création de base de donnée, requêtes SQL, mise en œuvre de déclencheurs base de données en PL/SQL. Ces trois parties utilisent le schéma de la base de données de production décrit ci-dessous, extrait du schéma plus complet de la base de données d'une banque. Cette base de données décrit les clients de la banque et les comptes dont ils sont titulaires.



La problématique de ce projet est d'utiliser les langages SQL et PL/SQL pour réaliser des requêtes SQL et écrire un programme PL/SQL pour actualiser automatiquement les informations dans la base de données.

1. Créer la base de données

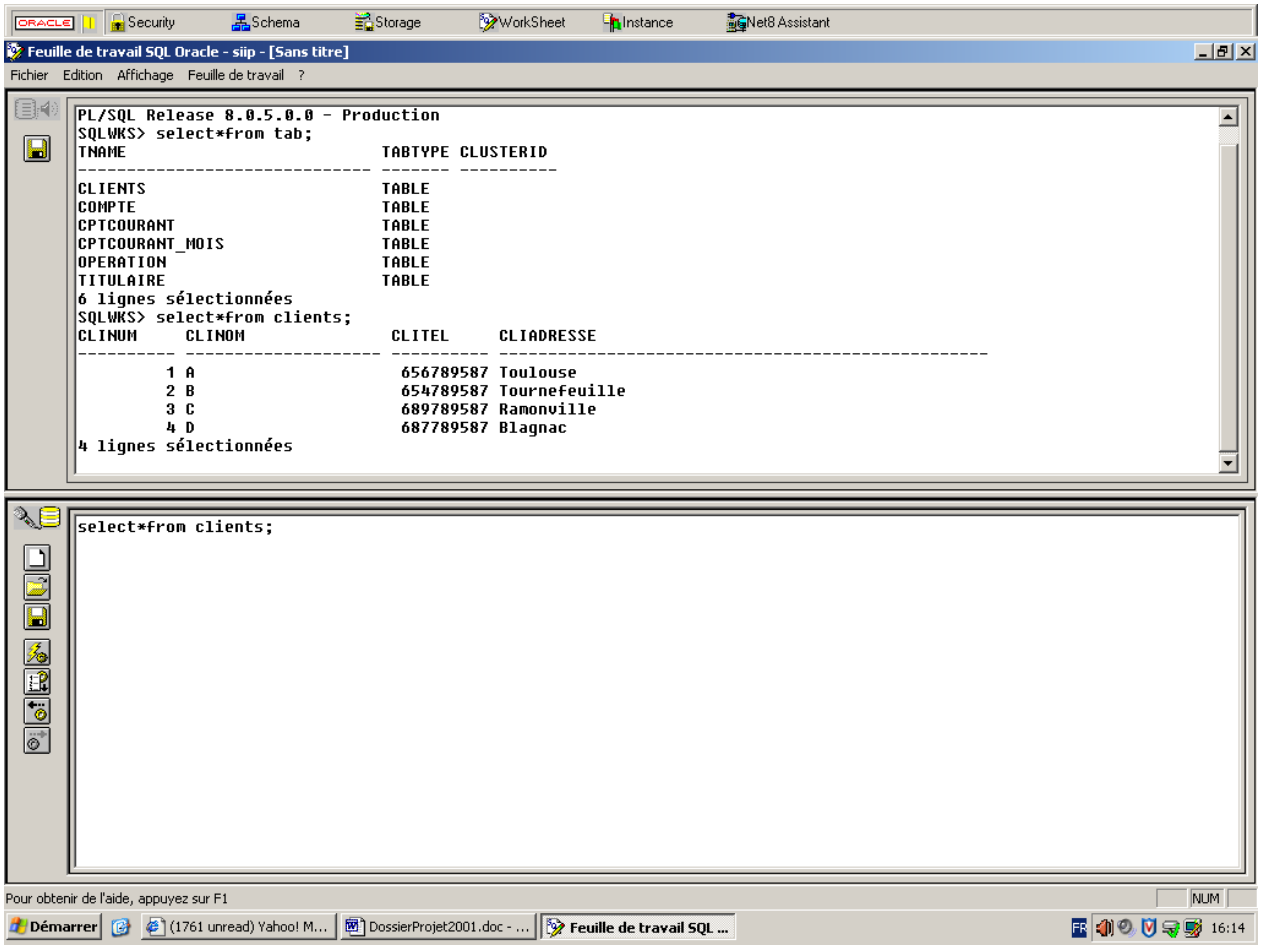
Pour créer la base de donnée on utilise le serveur SIIP. Ce serveur est localisé sur la machine nommée sip-app.univ-tlse1.fr. On a créé un utilisateur PLNPRJ en utilisant le client Security. Ensuite, il faut utiliser le client Worksheet pour exécuter le fichier. CREER_BANQUE.SQL qui contient les scripts SQL pour créer les tables et les contraintes de la base de données.

On va faire l'instanciation de la base de données.

1.1. Insérer des données pour la table clients

On a utilisé les scripts SQL comme suite pour insérer 4 clients :

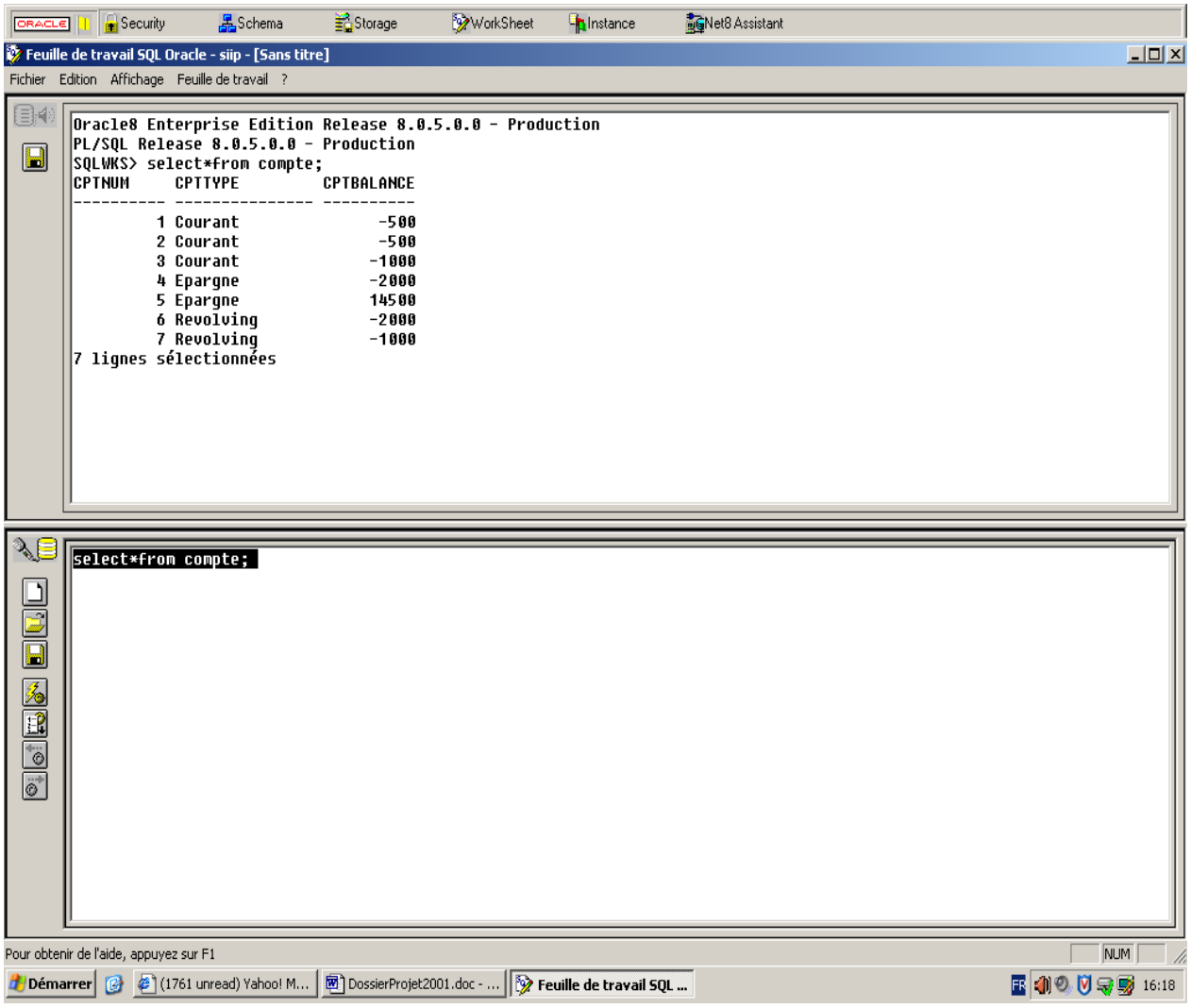
```
insert into clients values(1,'A',0656789587,'Toulouse');
insert into clients values(2,'B',0654789587,'Tournefeuille');
insert into clients values(3,'C',0689789587,'Ramonville');
insert into clients values(4,'D',0687789587,'Blagnac');
commit;
```



1.2. Insérer des données pour la table compte

On a utilisé les scripts SQL comme suite pour insérer 7 comptes:

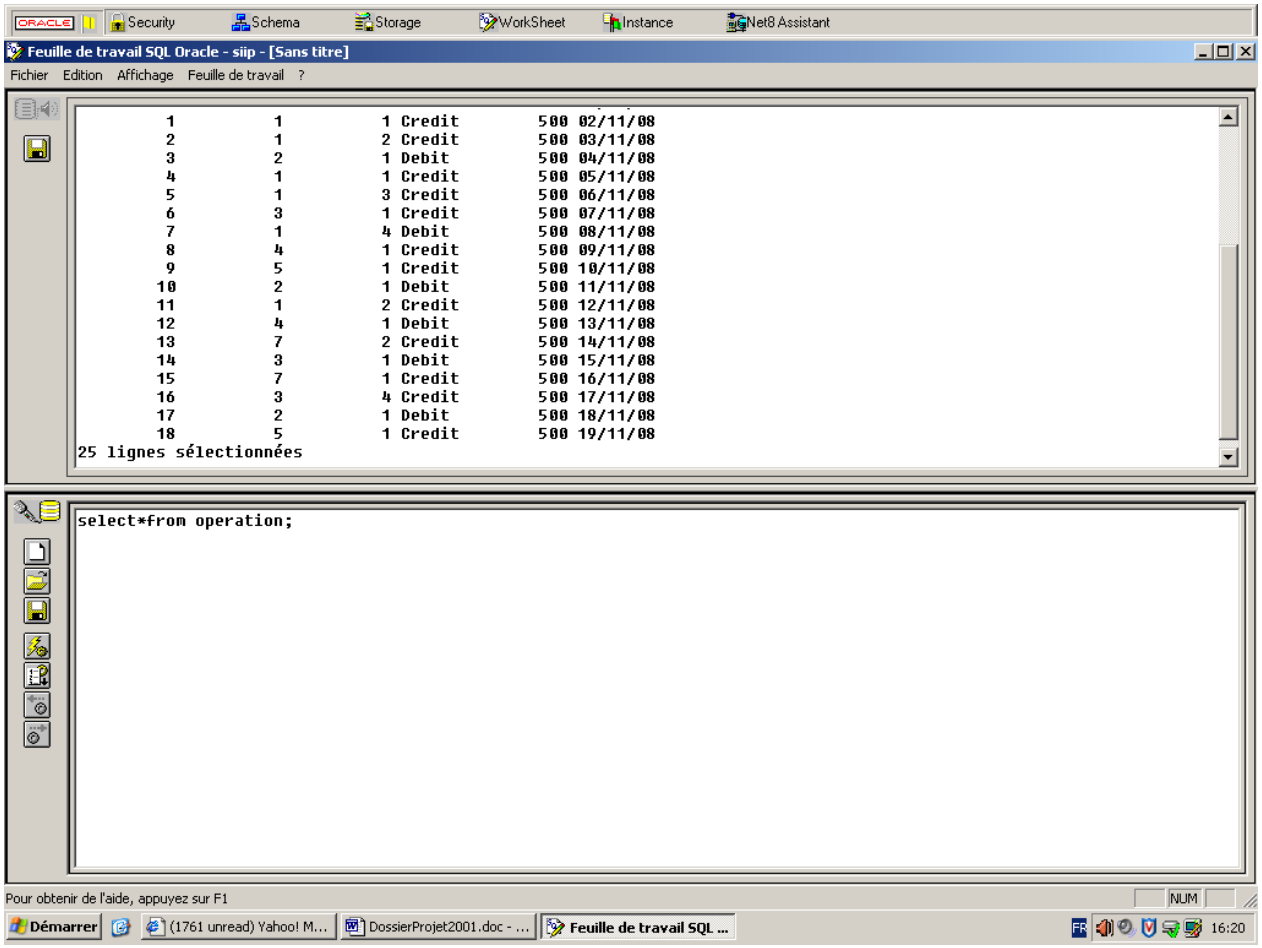
```
insert into compte values(1,'Courant',2000);
insert into compte values(2,'Courant',1000);
insert into compte values(3,'Courant',3000);
insert into compte values(4,'Epargne',4000);
insert into compte values(5,'Epargne',6000);
insert into compte values(6,'Revolving',2000);
insert into compte values(7,'Revolving',1000);
commit;
```



1.3. Insérer des données pour la table opération

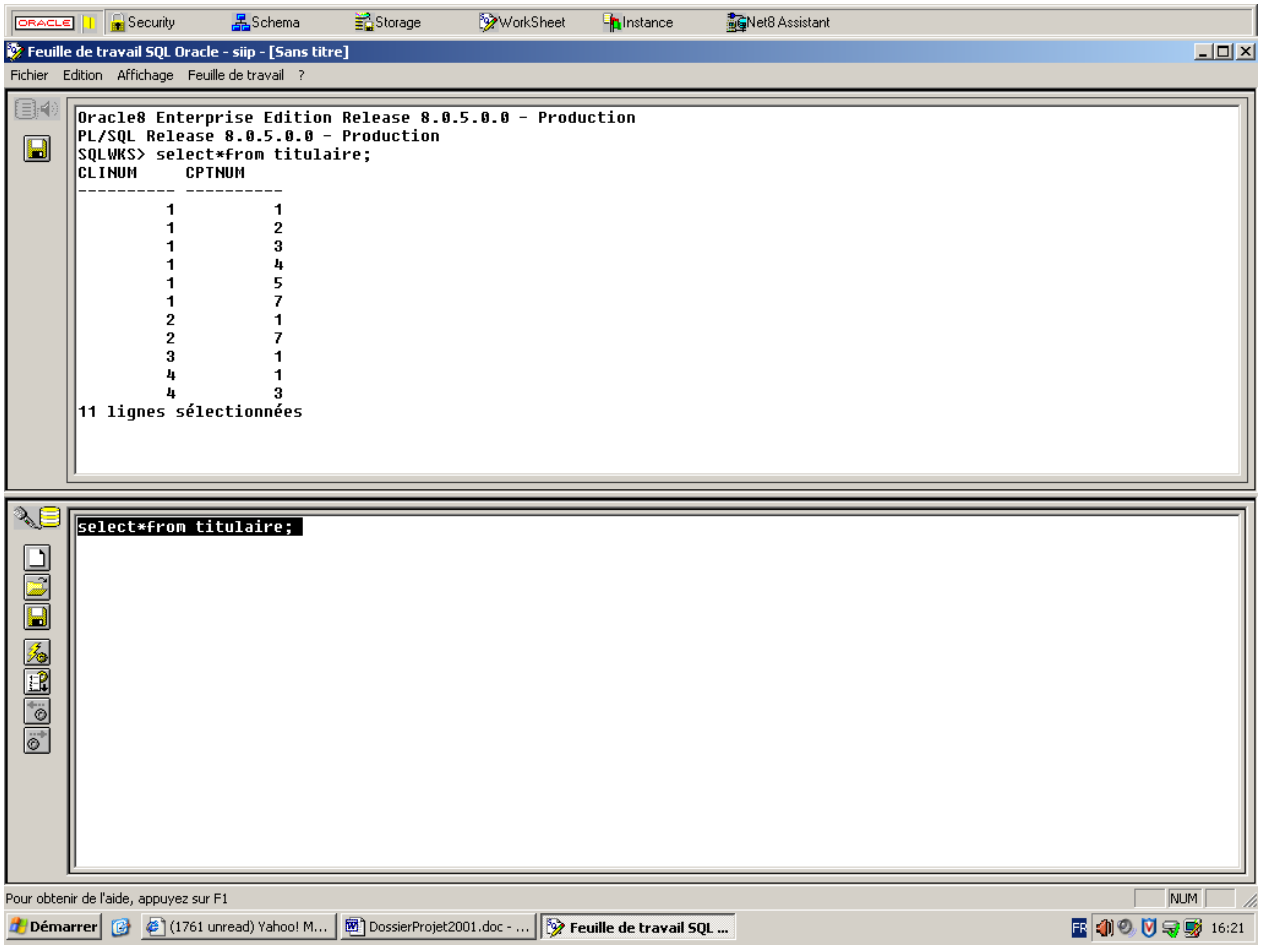
On a utilisé les scripts SQL comme suite pour insérer 18 opérations:

```
insert into operation values(1,1,1,'Credit',500,'02/11/08');
insert into operation values(2,1,2,'Credit',500,'03/11/08');
insert into operation values(3,2,1,'Debit',500,'04/11/08');
insert into operation values(4,1,1,'Credit',500,'05/11/08');
insert into operation values(5,1,3,'Credit',500,'06/11/08');
insert into operation values(6,3,1,'Credit',500,'07/11/08');
insert into operation values(7,1,4,'Debit',500,'08/11/08');
insert into operation values(8,4,1,'Credit',500,'09/11/08');
insert into operation values(9,5,1,'Credit',500,'10/11/08');
insert into operation values(10,2,1,'Debit',500,'11/11/08');
insert into operation values(11,1,2,'Credit',500,'12/11/08');
insert into operation values(12,4,1,'Debit',500,'13/11/08');
insert into operation values(13,7,2,'Credit',500,'14/11/08');
insert into operation values(14,3,1,'Debit',500,'15/11/08');
insert into operation values(15,7,1,'Credit',500,'16/11/08');
insert into operation values(16,3,4,'Credit',500,'17/11/08');
insert into operation values(17,2,1,'Debit',500,'18/11/08');
insert into operation values(18,5,1,'Credit',500,'19/11/08');
commit;
```



Pour remplir la table titulaire on a utilisé les scripts SQL comme suite :

insert into titulaire
select distinct clinum,cptnum from operation;



2. Les requêtes SQL

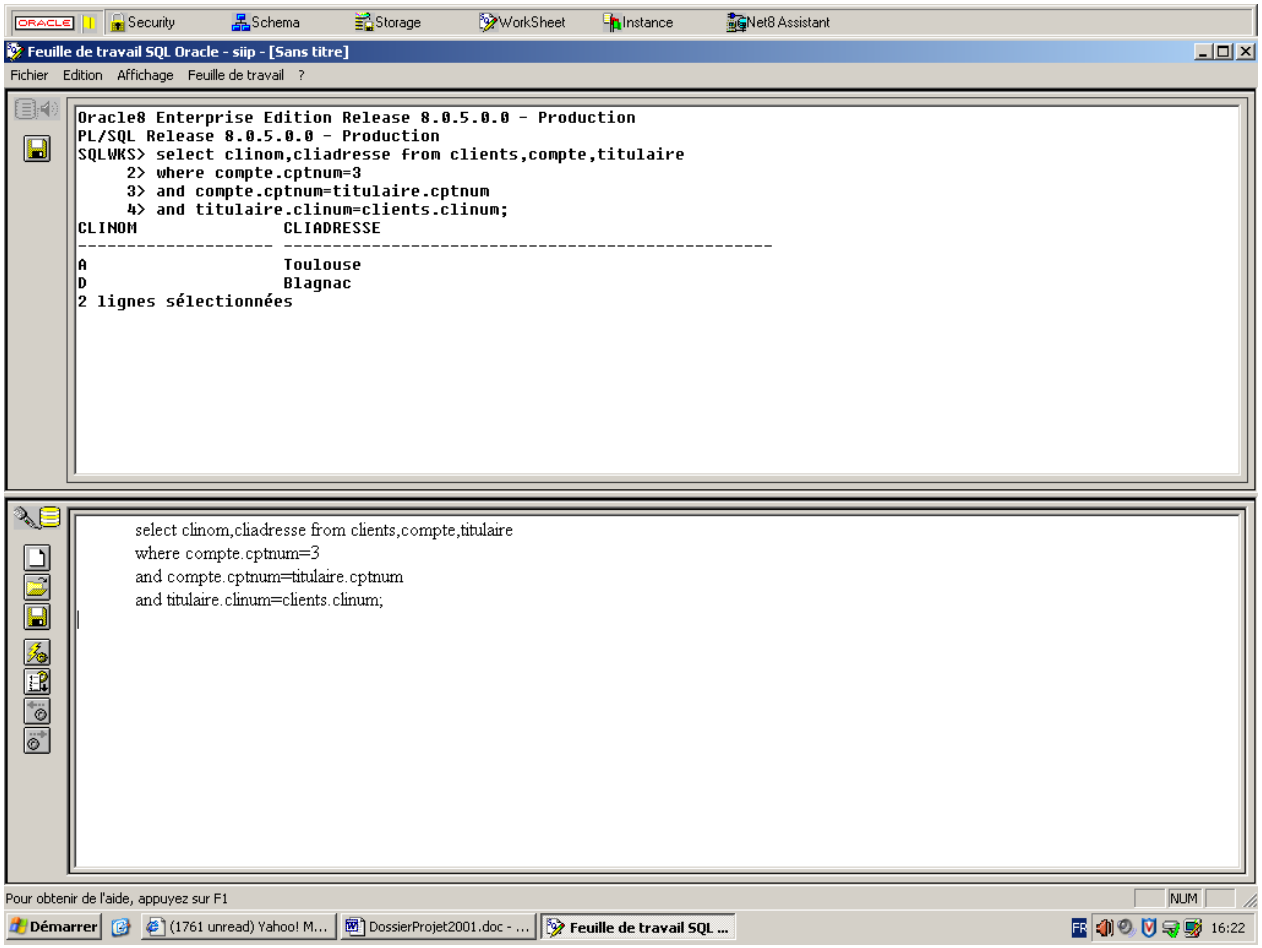
2.1. Afficher le nom et l'adresse de tous les titulaires du compte N° X

On a utilisé la requête SQL comme suite :

```
select clinom,cliadresse from clients,compte,titulaire
where compte.cptnum=3
and compte.cptnum=titulaire.cptnum
and titulaire.clinum=clients.clinum;
```

Or on peut aussi utiliser la requête comme suite :

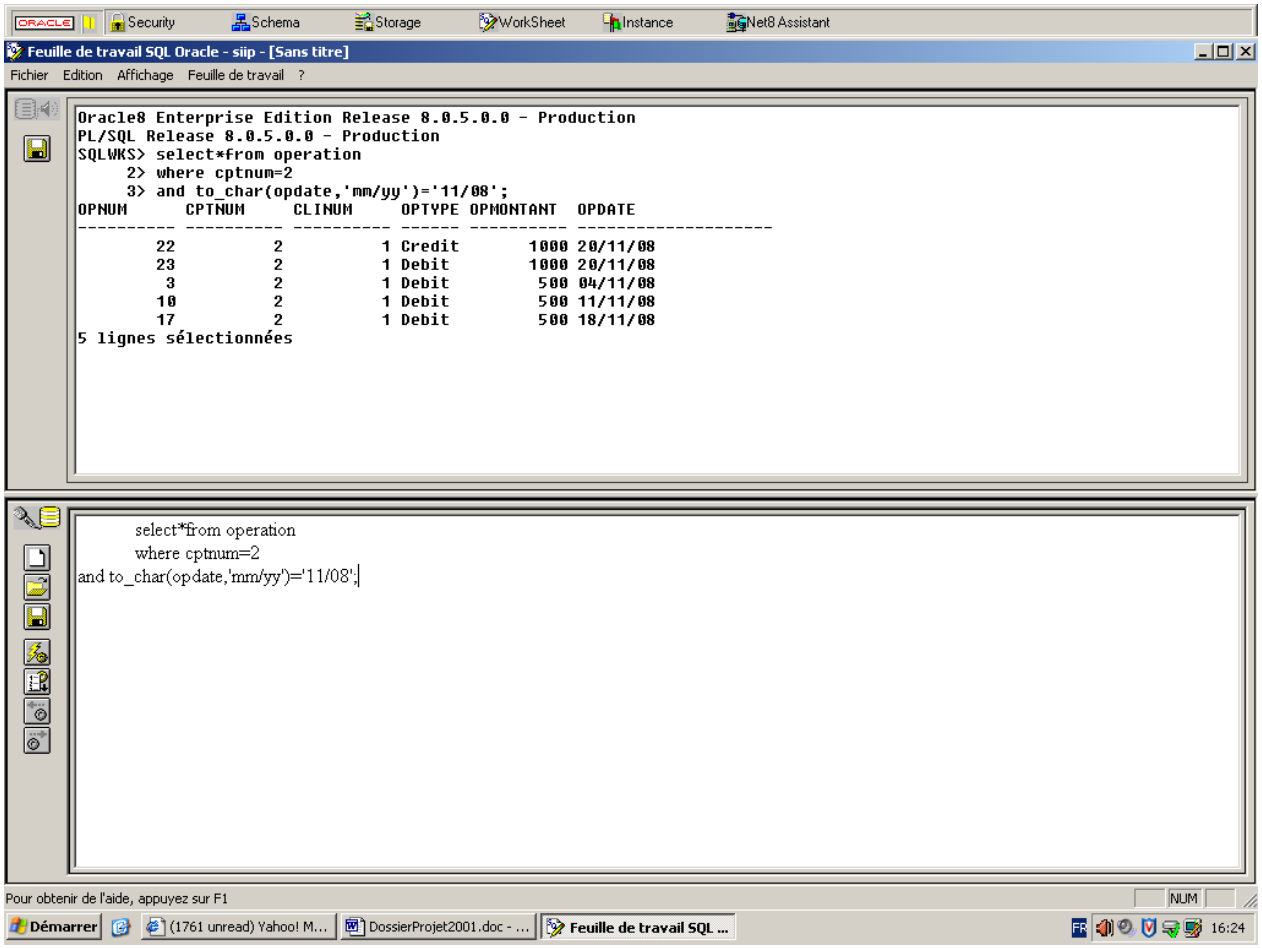
```
select distinct clinom,cliadresse from clients,compte,operation
where compte.cptnum=3
and compte.cptnum=operation.cptnum
and operation.clinum=clients.clinum;
```



2.2. Afficher la liste des opérations effectuées sur le compte N° X au cours du mois de novembre

On a utilisé la requête SQL avec la fonction to_char pour récupérer le mois:

```
select*from operation
where cptnum=2
and to_char(opdate,'mm/yy')='11/08';
```



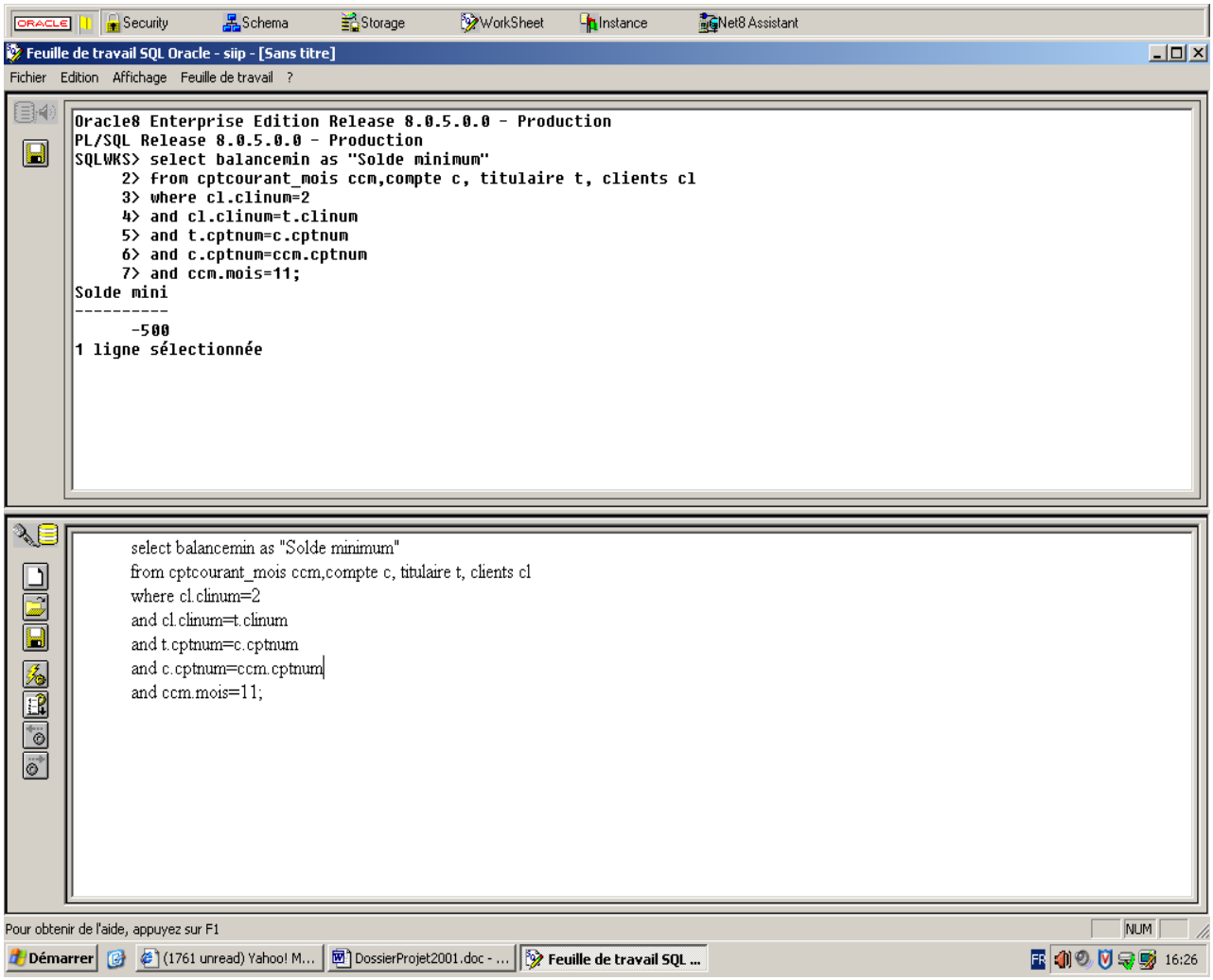
2.3. Afficher la valeur du solde minimum au mois de novembre, pour chaque compte dont Monsieur 'X' est titulaire

On a utilisé la requête SQL comme suite avec le numéro de client:

```
select balancemin as "Solde minimum"
from cptcourant_mois ccm,compte c, titulaire t, clients cl
where cl.clinum=2
and cl.clinum=t.clinum
and t.cptnum=c.cptnum
and c.cptnum=ccm.cptnum
and ccm.mois=11;
```

Or on peut aussi utiliser la requête comme suite avec le nom de client:

```
select balancemin as "Solde minimum"
from cptcourant_mois ccm,compte c, titulaire t, clients cl
where cl.clinom='A'
and cl.clinum=t.clinum
and t.cptnum=c.cptnum
and c.cptnum=ccm.cptnum
and ccm.mois=11;
```


ORACLE
Security
Schema
Storage
WorkSheet
Instance
Net8 Assistant
Feuille de travail SQL Oracle - siip - [Sans titre]
Fichier Edition Affichage Feuille de travail ?
Oracle8 Enterprise Edition Release 8.0.5.0.0 - Production
PL/SQL Release 8.0.5.0.0 - Production
SQLWKS> select balancemin as "Solde minimum"
2> from cptcourant_mois ccm,compte c, titulaire t, clients cl
3> where cl.clinum=2
4> and cl.clinum=t.clinum
5> and t.cptnum=c.cptnum
6> and c.cptnum=ccm.cptnum
7> and ccm.mois=11;
Solde mini

-500
1 ligne sélectionnée
select balancemin as "Solde minimum"
from cptcourant_mois ccm,compte c, titulaire t, clients cl
where cl.clinum=2
and cl.clinum=t.clinum
and t.cptnum=c.cptnum
and c.cptnum=ccm.cptnum
and ccm.mois=11;
Pour obtenir de l'aide, appuyez sur F1
NUM
Démarrer
(1761 unread) Yahoo! M...
DossierProjet2001.doc - ...
Feuille de travail SQL ...
16:26

3. Programmation PL/SQL

On a écrit le programme PL/SQL permettant d'actualiser automatiquement le solde d'un compte lors de l'enregistrement d'une opération, le programme PL/SQL permettant d'actualiser automatiquement la colonne DateDecouvert de la table CPTCOURANT et les colonnes BalanceMin et BalanceMax de la table CPTCOURANT_MOIS lors de la mise à jour du solde d'un compte courant.

On a fait les 3 traitements dans un seul déclencheur sur la table opération et on a crée les déclencheurs sur un seul, afin que les deux derniers déclencheurs interviennent après mise à jour du solde du compte pour récupérer le nouveau solde, car on a constaté qu'en créant séparément notamment le déclencheur de la mise à jour de balancemin et balancemax, la mise à jour se faisait par rapport à l'ancien solde et pas le nouveau.

Ce déclencheur intervient après Insertion d'une nouvelle opération et c'est pour une mise à jour.

Le code comme est suit :

```
Declare
    nombrel cptcourant_mois.cptnum%type;
    nombrel2 cptcourant.cptnum%type;
    typecpt compte.cpttype%type;
    montant compte.cptbalance%type;
    soldemin cptcourant_mois.balancemin%type;
    soldemax cptcourant_mois.balancemax%type;
    dd cptcourant.datedecouvert%type;

Begin
    -- Programme PL/SQL permettant d'actualiser automatiquement le solde d'un compte
    lors de l'enregistrement d'une opération

    if :n.optype='Credit' then
    update compte
    set cptbalance=cptbalance+:n.opmontant
    where cptnum=:n.cptnum;
    else
    update compte
    set cptbalance=cptbalance-:n.opmontant
    where cptnum=:n.cptnum;
    end if;

    --Récupérer le type du compte on le mettant dans une variable
    select cpttype into typecpt from compte
    where cptnum=:n.cptnum;
    --Récupérer le montant du solde du compte en le mettant dans une variable
    select cptbalance into montant from compte
    where cptnum=:n.cptnum;

    if typecpt='Courant' then
```

-- Programme PL/SQL permettant d'actualiser automatiquement la colonne DateDecouvert de la table CPTCOURANT

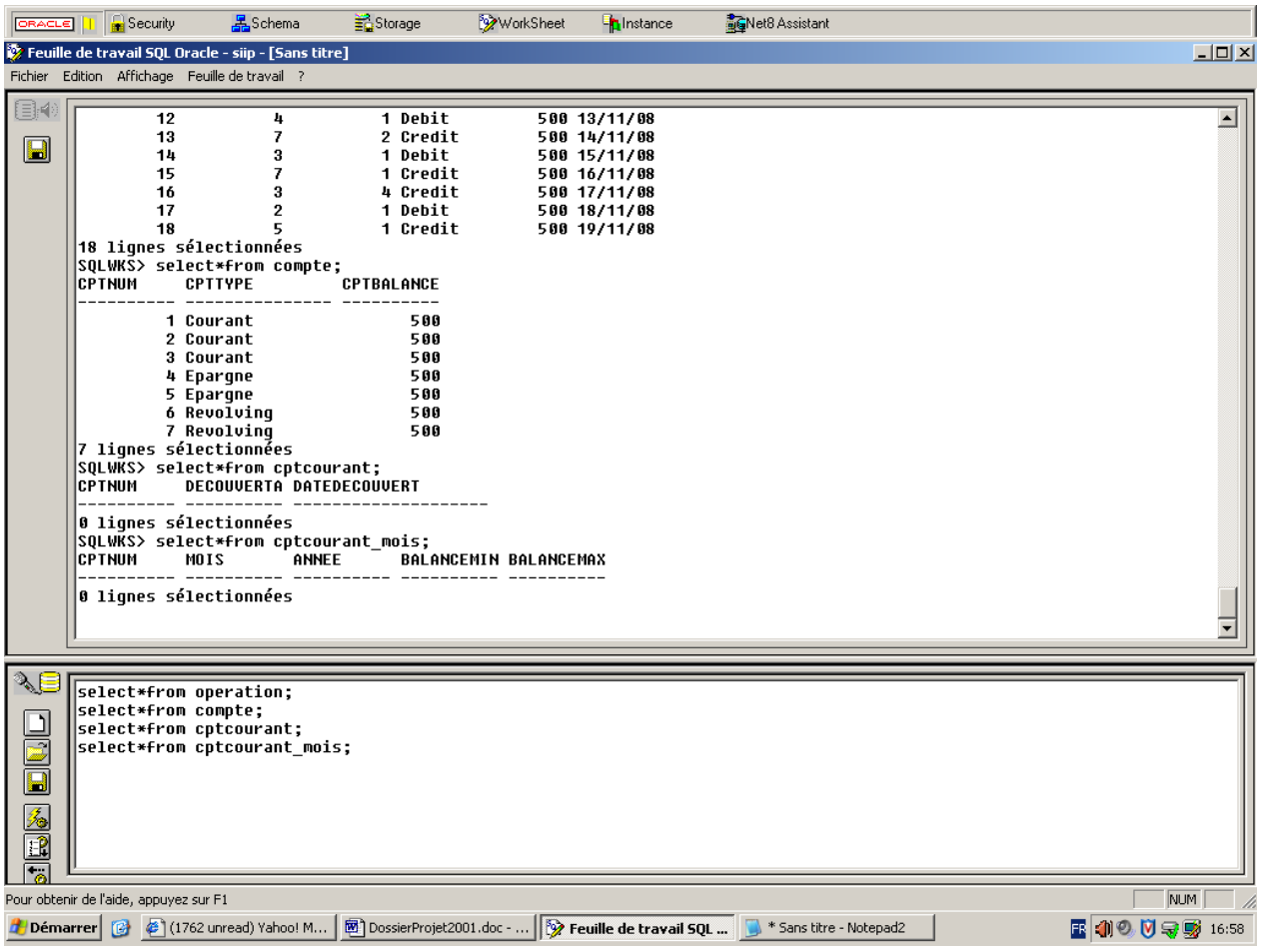
```
	--Vérifier l'existence d'un compte dans la table cptcourant
select count(cptnum) into nombrel2 from cptcourant
where cptnum=:n.cptnum;

if nombrel2!=0 then
	--Récupérer la date de découvert en la mettant dans une variable
	select datedecouvert into dd from cptcourant
	where cptnum=:n.cptnum;
	if dd is null then
		 if montant<0 then
			update cptcourant
			set datedecouvert=:n.opdate
			where cptnum=:n.cptnum;
		end if;
	else
		if montant>=0 then
			update cptcourant
			set datedecouvert=null
			where cptnum=:n.cptnum;
		end if;
	end if;
else
	if montant<0 then
		insert into cptcourant
		values(:n.cptnum,500,:n.opdate);
	else
		insert into cptcourant
		values(:n.cptnum,500,null);
	end if;
end if;
```

-- Programme PL/SQL permettant d'actualiser automatiquement les colonnes BalanceMin et BalanceMax de la table CPTCOURANT_MOIS lors de la mise à jour du solde d'un compte courant.

```
--Vérifier l'existence d'un compte dans la table cptcourant_mois
	select count(cptnum) into nombrel from cptcourant_mois
where cptnum=:n.cptnum
and mois=to_char(:n.opdate,'mm')
and annee=to_char(:n.opdate,'yyyy');

if nombrel=0 then
	insert into cptcourant_mois
	values(:n.cptnum,to_char(:n.opdate,'mm'),
		to_char(:n.opdate,'yyyy'),montant,montant);
else
		--Récupérer les soldes minimum et maximum en les mettant dans des
		variables
		select balancemin,balancemax into soldemin,soldemax from
		cptcourant_mois
	where cptnum=:n.cptnum
```

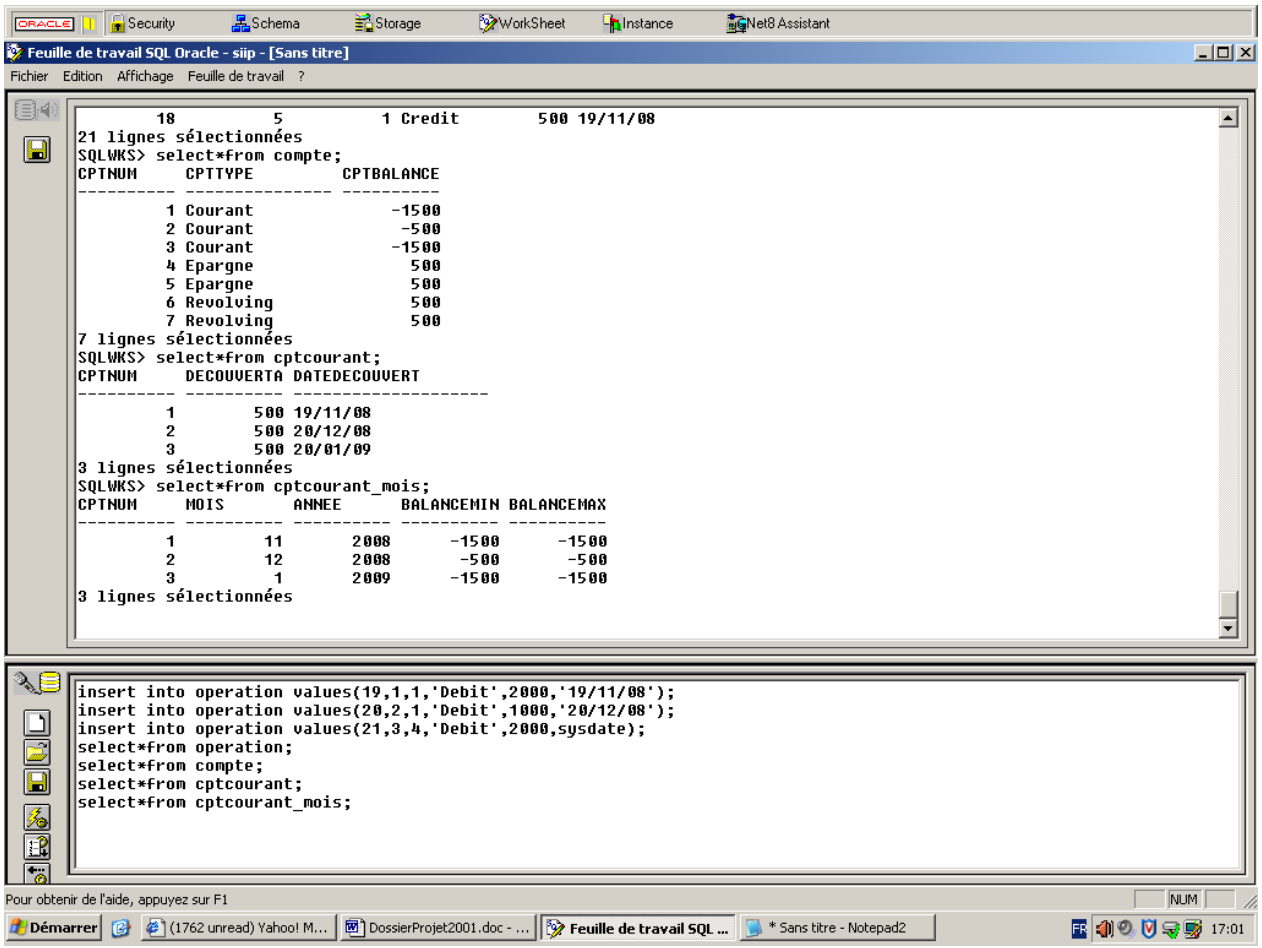
```
                and mois=to_char(:n.opdate,'mm')
                and annee=to_char(:n.opdate,'yyyy');

                if montant<soldemin then
                        update cptcourant_mois
                        set balancemin=montant
                        where cptnum=:n.cptnum
                        and mois=to_char(:n.opdate,'mm')
                        and annee=to_char(:n.opdate,'yyyy');
                end if;
                if montant>soldemax then
                        update cptcourant_mois
                        set balancemax=montant
                        where cptnum=:n.cptnum
                        and mois=to_char(:n.opdate,'mm')
                        and annee=to_char(:n.opdate,'yyyy');
                end if;
          end if;
        end if;
end;
```

La base de donnés avant d'insérer une nouvelle opération :

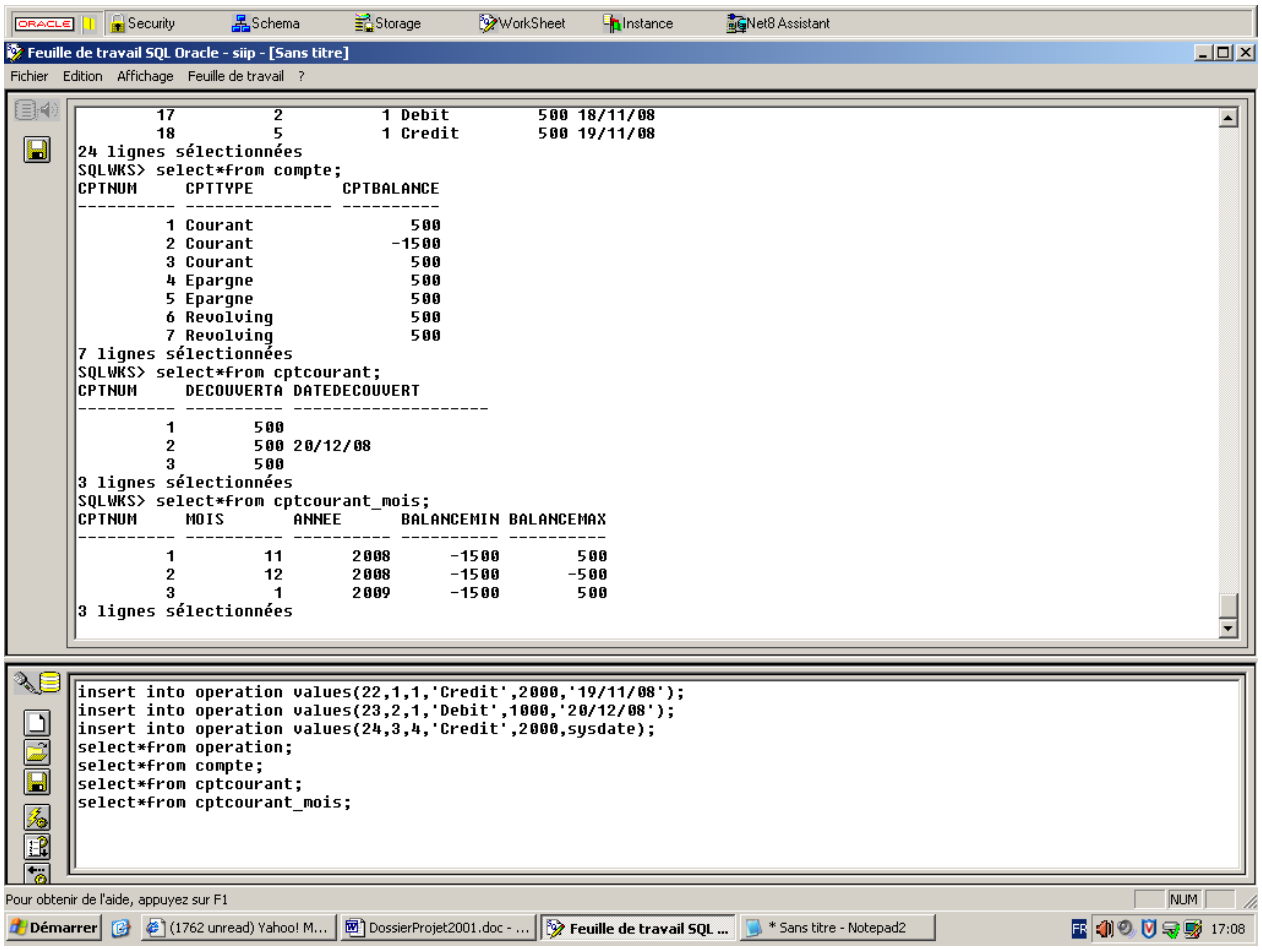


Base de données après insertion opérations:

Comme les deux tables cptcourant et cptcourant_mois sont vides donc le « select count=0 ».



Ici comme les tables cptcourant et cptcourant_mois ont des données, donc le « select count » est différent de 0.



Conclusion

Nous pensons que nous avons accomplit note projet avec un petits succès. Nous avons bien cerné le projet en faisant beaucoup d'efforts. Grâce au projet, on a élargi nos connaissances notamment en travaillant en équipe.

On a rédigé notre projet en expliquant clairement la relation entre le système d'information décisionnel et la vie pratique, qui est dans ce cas l'activité d'une banque et ses clients. Il existe encore des améliorations à effectuer. On espère que notre travaille saura vous satisfaire.

En fin, nous vous remercions beaucoup de votre attention et de nous excuser pour les quelques erreurs qui peuvent exister.