                 

Manuel d’utilisation

Rédigé par Ion Baltaga, Baptiste Lacroix, Ilyane Selmane, Alexandre Colin

       Sous la direction de Lise Brenac, Daniel Portuguais, Dijana Bojovic

Année universitaire : 2021/2022



Indicateurs

Indicateur 1 :

/\*\*  
 Requête permettant de connaitre le nombre d'utilisateurs total

Depuis la création du site  
 \*/

SELECT COUNT(\*)  
FROM UTILISATEUR;

*Pour exécuter cette requête il suffit de la sélectionner puis de cliquer sur cette icône :*

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Requête permettant de connaitre le nombre d'utilisateur particulier total  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM UTILISATEURPARTICULIER;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 3 :

/\*\*  
 Requête permettant de connaitre le nombre d'associations total  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM UTILISATEURASSOCIATION;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 4 :

/\*\*  
 Requête permettant de connaitre le nombre d'entrepreneur total  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM UTILISATEURENTREPRENEUR;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 5 :

/\*\*  
 Requête permettant de connaitre le nombre d'entreprises total  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM UTILISATEURENTREPRISE;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 6 :

/\*\*  
 Requête permettant de connaitre le nombre de demande total effectué  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM DEMANDE;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 7 :

/\*\*  
 Requête permettant de connaitre le nombre de service total proposés  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM SERVICES;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 8 :

/\*\*  
 Requête permettant de connaitre le nombre de bien total proposés  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM BIEN;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 9 :

/\*\*  
 Requête permettant de connaitre le nombre de publicité total publiés  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM PUBLICITE;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 10 :

/\*\*  
 Requête permettant de connaitre le nombre d'utilisateurs étants abonnés au news  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM UTILISATEUR  
WHERE ABONNEMENT = 1;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 11 :

/\*\*  
 Requête permettant de connaitre le nombre total de vendeurs  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM VENDEUR;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 12 :

/\*\*  
 Requête permettant de connaitre le nombre total de clients  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM CLIENTS;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 13 :

/\*\*  
 Requête permettant de connaitre le nombre de demande faites depuis la création du site  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM DEMANDE;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 14 :

/\*\*  
 Requête permettant de connaitre le nombre d'avis créés depuis la création du site  
 Depuis la création du site  
 \*/  
SELECT COUNT(\*)  
FROM AVIS;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 15 :

/\*\*  
 Requête permettant de connaitre pour tous les utilisateurs le nombres d'avis reçus  
 Depuis la création du site  
 \*/  
select idUtilisateur, nbrAvisTotal  
from utilisateur  
group by idUtilisateur, nbrAvisTotal;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Requête permettant de connaître le nombre d'utilisateurs pour chaque ville  
 \*/  
select count(\*) as utilisateurs, UTILISATEUR.CODEPOSTAL  
from utilisateur  
group by CODEPOSTAL;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Requête permettant d'afficher le nombre d'utilisateurs entrepreneurs classé par metier  
 \*/  
select count(\*) as utilisateurs\_Entrepreneurs, metier  
from utilisateurentrepreneur  
group by metier;

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 View qui permet de voir le nombres de demandes par code postaux  
 \*/  
CREATE OR REPLACE VIEW demande\_par\_zone\_geographique AS  
SELECT CODEPOSTAL AS codeP, COUNT(\*) AS total  
FROM DEMANDE  
 INNER JOIN UTILISATEUR U on U.IDUTILISATEUR = DEMANDE.IDUTILISATEUR  
GROUP BY U.CODEPOSTAL;  
  
SELECT \*  
FROM demande\_par\_zone\_geographique;

*Ici pour pouvoir accéder à la vue il faut dans un premier temps exécuter la vue et ensuite le select comme expliqué précédemment*

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Procédure permettant d'afficher la moyenne des notes en fonction de l'input de l'utilistaeur  
 @param : input -> type NUMBER  
 Le numéro rentré en paramère correspond aux données auxquelles l'utilisateur souhaite accéder  
 \*/

CREATE OR REPLACE PROCEDURE *affiche\_moyenne\_notes*(input NUMBER)  
AS  
 badValue exception;  
 total FLOAT;  
 numerateur FLOAT;  
 denominateur FLOAT;  
BEGIN  
 CASE TO\_NUMBER(input)  
 WHEN 1 THEN SELECT SUM(UTILISATEUR.MOYENNENOTES) into numerateur FROM UTILISATEUR;  
 SELECT COUNT(\*) into denominateur FROM UTILISATEUR;  
 total := numerateur / denominateur;  
 dbms\_output.*PUT\_LINE*('La moyenne des notes de tous les utilisateurs est de : ' || total);  
 WHEN 2 THEN SELECT SUM(UTILISATEUR.MOYENNENOTES)  
 into numerateur  
 FROM UTILISATEUR  
 INNER JOIN UTILISATEURENTREPRISE  
 ON UTILISATEUR.IDUTILISATEUR = UTILISATEURENTREPRISE.IDUTILISATEURENTREPRISE;  
 SELECT COUNT(\*) into denominateur FROM UTILISATEUR;  
 total := numerateur / denominateur;  
 dbms\_output.*PUT\_LINE*('La moyenne des notes de toutes les entreprises est de : ' || total);  
 WHEN 3 THEN SELECT SUM(UTILISATEUR.MOYENNENOTES)  
 into numerateur  
 FROM UTILISATEUR  
 INNER JOIN UTILISATEURASSOCIATION U  
 on UTILISATEUR.IDUTILISATEUR = U.IDUTILISATEURASSOCIATION;  
 SELECT COUNT(\*) into denominateur FROM UTILISATEUR;  
 total := numerateur / denominateur;  
 dbms\_output.*PUT\_LINE*('La moyenne des notes de toutes les associations est de : ' || total);  
 WHEN 4 THEN SELECT SUM(UTILISATEUR.MOYENNENOTES)  
 into numerateur  
 FROM UTILISATEUR  
 INNER JOIN UTILISATEURENTREPRENEUR U2  
 on UTILISATEUR.IDUTILISATEUR = U2.IDUTILISATEURENTREPRENEUR;  
 SELECT COUNT(\*) into denominateur FROM UTILISATEUR;  
 total := numerateur / denominateur;  
 dbms\_output.*PUT\_LINE*('La moyenne des notes de tous les entrepreneurs est de : ' || total);  
 WHEN 5 THEN SELECT SUM(UTILISATEUR.MOYENNENOTES)  
 into numerateur  
 FROM UTILISATEUR  
 INNER JOIN UTILISATEURPARTICULIER U3  
 on UTILISATEUR.IDUTILISATEUR = U3.IDUTILISATEURPARTICULIER;  
 SELECT COUNT(\*) into denominateur FROM UTILISATEUR;  
 total := numerateur / denominateur;  
 dbms\_output.*PUT\_LINE*('La moyenne des notes de tous les particuliers est de : ' || total);  
 ELSE RAISE badValue;  
 END CASE;  
EXCEPTION  
 WHEN badValue THEN dbms\_output.*PUT\_LINE*('Le numéro donné ne correspond pas !');  
 WHEN others THEN dbms\_output.*PUT\_LINE*('Erreur !');  
end;  
/  
  
/\*\*  
 Fonction permettant de savoir si la chaine entrée est un numéro  
 \*/  
CREATE OR REPLACE FUNCTION *Is\_Number*(P\_String IN VARCHAR2) RETURN BOOLEAN IS  
 v\_number NUMBER;  
BEGIN  
 v\_number := to\_number(P\_String);  
 RETURN (TRUE);  
EXCEPTION  
 WHEN VALUE\_ERROR THEN  
 RETURN (FALSE);  
END;  
/  
  
BEGIN  
 /\*\*  
 Si vous voulez la moyenne de tous les utilisateurs entrez (1)  
 Si vous voulez la moyenne de toutes les entreprises entrez (2)  
 Si vous voulez la moyenne de toutes les associations entrez (3)  
 Si vous voulez la moyenne de tous les entrepreneurs entrez (4)  
 Si vous voulez la moyenne de tous les particuliers entrez (5)  
 \*/  
 *affiche\_moyenne\_notes*(1);  
end;  
/

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Procédure permettant de savoir le nombres d'utilisateurs se situant dans le code postal rentré en paramètre  
 \*/  
CREATE OR REPLACE PROCEDURE *affiche\_nbr\_pour\_region*(codeP LOCALISATION.CODEPOSTAL%type)  
AS  
 ex\_code\_invalide exception;  
 ex\_code\_trop\_long exception;  
 pragma exception\_init ( ex\_code\_trop\_long, -6502 );  
 total INTEGER;  
BEGIN  
 IF codeP = TO\_NUMBER(codeP) AND LENGTH(codeP) = 5 THEN  
 select count(\*) into total from localisation where codePostal like codeP;  
 DBMS\_OUTPUT.*PUT\_LINE*('Il y a ' || total || ' utilisateurs se situant dans la région : ' || codeP);  
 ELSE  
 RAISE ex\_code\_invalide;  
 END IF;  
EXCEPTION  
 WHEN ex\_code\_trop\_long THEN dbms\_output.*put\_line*('Code postal trop long !');  
 WHEN ex\_code\_invalide THEN dbms\_output.*put\_line*('Code postal invalide !');  
 WHEN others THEN dbms\_output.*put\_line*('Erreur !');  
END;  
/  
  
BEGIN  
 *affiche\_nbr\_pour\_region*('57340');  
end;  
/

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Affiche la ou les categories les moins utilisées  
 \*/  
CREATE OR REPLACE PROCEDURE *categorie\_moins\_utilises* IS  
 temp integer;  
 temp\_bien integer;  
 temp\_service integer;  
 min\_categorie integer := 9999;  
 nom\_categorie SERVICES.nom%TYPE;  
 number\_categorie integer;  
BEGIN  
 SELECT COUNT(\*) INTO number\_categorie FROM CATEGORIE;  
 for i in 1..number\_categorie  
 LOOP  
 temp := 0;  
 SELECT COUNT(\*) into temp\_bien FROM BIEN WHERE IDCATEGORIE = i;  
 SELECT COUNT(\*) into temp\_service FROM SERVICES WHERE IDCATEGORIE = i;  
 temp := temp\_bien + temp\_service;  
 IF temp < min\_categorie THEN  
 min\_categorie := temp;  
 end if;  
 end loop;  
  
 for i in 1..number\_categorie  
 LOOP  
 temp := 0;  
 SELECT COUNT(\*) into temp\_bien FROM BIEN WHERE IDCATEGORIE = i;  
 SELECT COUNT(\*) into temp\_service FROM SERVICES WHERE IDCATEGORIE = i;  
 temp := temp\_bien + temp\_service;  
 IF temp = min\_categorie THEN  
 SELECT CATEGORIE.NOM INTO nom\_categorie FROM CATEGORIE WHERE IDCATEGORIE = i;  
 DBMS\_OUTPUT.*PUT\_LINE*('La catégorie la moins utilisée est : ' || nom\_categorie);  
 end if;  
 end loop;  
end;  
/  
  
BEGIN  
 *categorie\_moins\_utilises*();  
end;  
/

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Affiche la ou les categories les plus utilisées  
 \*/  
CREATE OR REPLACE PROCEDURE *categorie\_plus\_utilises* IS  
 temp integer;  
 temp\_bien integer;  
 temp\_service integer;  
 max\_categorie integer := 0;  
 nom\_categorie SERVICES.nom%TYPE;  
 number\_categorie integer;  
BEGIN  
 SELECT COUNT(\*) INTO number\_categorie FROM CATEGORIE;  
 for i in 1..number\_categorie  
 LOOP  
 temp := 0;  
 SELECT COUNT(\*) into temp\_bien FROM BIEN WHERE IDCATEGORIE = i;  
 SELECT COUNT(\*) into temp\_service FROM SERVICES WHERE IDCATEGORIE = i;  
 temp := temp\_bien + temp\_service;  
 IF temp > max\_categorie THEN  
 max\_categorie := temp;  
 end if;  
 end loop;  
  
 for i in 1..number\_categorie  
 LOOP  
 temp := 0;  
 SELECT COUNT(\*) into temp\_bien FROM BIEN WHERE IDCATEGORIE = i;  
 SELECT COUNT(\*) into temp\_service FROM SERVICES WHERE IDCATEGORIE = i;  
 temp := temp\_bien + temp\_service;  
 IF temp = max\_categorie THEN  
 SELECT CATEGORIE.NOM INTO nom\_categorie FROM CATEGORIE WHERE IDCATEGORIE = i;  
 DBMS\_OUTPUT.*PUT\_LINE*('La/Les catégories la/les plus utilisée(s) est/sont : ' || nom\_categorie);  
 end if;  
 end loop;  
end;  
/  
  
BEGIN  
 *categorie\_plus\_utilises*();  
end;  
/

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Affiche la ou les utilisateurs ayants les moins bonnes notations  
 \*/  
CREATE OR REPLACE PROCEDURE *utilisateurs\_moins\_bien\_notes* IS  
 temp integer;  
 exist INTEGER;  
 min\_utilisateur integer := 9999;  
 id\_utilisateur UTILISATEUR.IDUTILISATEUR%TYPE;  
 number\_avis integer;  
 numerateur FLOAT;  
 denominateur FLOAT;  
BEGIN  
 SELECT COUNT(\*) INTO number\_avis FROM AVIS;  
 for i in 1..number\_avis  
 LOOP  
 temp := 0;  
 SELECT count(\*) into exist FROM AVIS WHERE IDAVIS = i;  
 IF exist > 0 THEN  
 SELECT SUM(AVIS.NOTEUTILISATEUR) into numerateur FROM AVIS WHERE IDAVIS = i GROUP BY AVIS.IDUTILISATEUR;  
 SELECT COUNT(\*) into denominateur FROM AVIS WHERE IDAVIS = i GROUP BY AVIS.IDUTILISATEUR;  
 temp := numerateur + denominateur;  
 IF temp < min\_utilisateur THEN  
 min\_utilisateur := temp;  
 end if;  
 end if;  
 end loop;  
  
 for i in 1..number\_avis  
 LOOP  
 temp := 0;  
 SELECT count(\*) into exist FROM AVIS WHERE IDAVIS = i;  
 IF exist > 0 THEN  
 SELECT SUM(AVIS.NOTEUTILISATEUR) into numerateur FROM AVIS WHERE IDAVIS = i GROUP BY AVIS.IDUTILISATEUR;  
 SELECT COUNT(\*) into denominateur FROM AVIS WHERE IDAVIS = i GROUP BY AVIS.IDUTILISATEUR;  
 temp := numerateur + denominateur;  
 IF temp = min\_utilisateur THEN  
 SELECT UTILISATEUR.IDUTILISATEUR  
 INTO id\_utilisateur  
 FROM UTILISATEUR  
 INNER JOIN AVIS A2 on UTILISATEUR.IDUTILISATEUR = A2.IDUTILISATEUR  
 WHERE A2.IDAVIS = i  
 GROUP BY A2.IDUTILISATEUR, UTILISATEUR.IDUTILISATEUR;  
 DBMS\_OUTPUT.*PUT\_LINE*('L"id ou les id utilisateur(s) ayant la/les moins bonne notation(s) sont : ' ||  
 id\_utilisateur);  
 end if;  
 end if;  
 end loop;  
end;  
/  
  
BEGIN  
 *utilisateurs\_moins\_bien\_notes*();  
end;  
/

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Affiche la ou les utilisateurs ayants les meilleurs notations  
 \*/  
CREATE OR REPLACE PROCEDURE *utilisateurs\_mieux\_notes* IS  
 temp integer;  
 exist INTEGER;  
 min\_utilisateur integer := -1;  
 id\_utilisateur UTILISATEUR.IDUTILISATEUR%TYPE;  
 number\_avis integer;  
 numerateur FLOAT;  
 denominateur FLOAT;  
BEGIN  
 SELECT COUNT(\*) INTO number\_avis FROM AVIS;  
 for i in 1..number\_avis  
 LOOP  
 temp := 0;  
 SELECT count(\*) into exist FROM AVIS WHERE IDAVIS = i;  
 IF exist > 0 THEN  
 SELECT SUM(AVIS.NOTEUTILISATEUR) into numerateur FROM AVIS WHERE IDAVIS = i GROUP BY AVIS.IDUTILISATEUR;  
 SELECT COUNT(\*) into denominateur FROM AVIS WHERE IDAVIS = i GROUP BY AVIS.IDUTILISATEUR;  
 temp := numerateur + denominateur;  
 IF temp > min\_utilisateur THEN  
 min\_utilisateur := temp;  
 end if;  
 end if;  
 end loop;  
  
 for i in 1..number\_avis  
 LOOP  
 temp := 0;  
 SELECT count(\*) into exist FROM AVIS WHERE IDAVIS = i;  
 IF exist > 0 THEN  
 SELECT SUM(AVIS.NOTEUTILISATEUR) into numerateur FROM AVIS WHERE IDAVIS = i GROUP BY AVIS.IDUTILISATEUR;  
 SELECT COUNT(\*) into denominateur FROM AVIS WHERE IDAVIS = i GROUP BY AVIS.IDUTILISATEUR;  
 temp := numerateur + denominateur;  
 IF temp = min\_utilisateur THEN  
 SELECT UTILISATEUR.IDUTILISATEUR  
 INTO id\_utilisateur  
 FROM UTILISATEUR  
 INNER JOIN AVIS A2 on UTILISATEUR.IDUTILISATEUR = A2.IDUTILISATEUR  
 WHERE A2.IDAVIS = i  
 GROUP BY A2.IDUTILISATEUR, UTILISATEUR.IDUTILISATEUR;  
 DBMS\_OUTPUT.*PUT\_LINE*('L"id ou les id utilisateur(s) ayant la/les meilleurs notation(s) sont : ' ||  
 id\_utilisateur);  
 end if;  
 end if;  
 end loop;  
end;  
/  
  
BEGIN  
 *utilisateurs\_mieux\_notes*();  
end;  
/

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Procedure pour donner le nombres de vente de biens en fonction du code postal  
 \*/  
create or replace procedure *affiche\_biens\_localisation*(code\_postal LOCALISATION.CODEPOSTAL%type)  
 is  
 id\_localisation int;  
 nb\_ventes int;  
begin  
 select idlocalisation into id\_localisation from localisation where code\_postal = codepostal;  
 select count(\*) into nb\_ventes from bien where idlocalisation = id\_localisation;  
 dbms\_output.*put\_line*('Il y a ' || nb\_ventes || ' vente(s).');  
EXCEPTION  
 WHEN no\_data\_found THEN DBMS\_OUTPUT.*PUT\_LINE*('Code postal inexistant !');  
 WHEN others THEN DBMS\_OUTPUT.*PUT\_LINE*('Erreur !');  
end;  
/  
  
BEGIN  
 *affiche\_biens\_localisation*('57340');  
end;  
/

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Procedure pour donner le nombres de propositions de services en fonction du code postal  
 \*/  
create or replace procedure *affiche\_services\_localisation*(code\_postal LOCALISATION.CODEPOSTAL%type)  
 is  
 id\_localisation int;  
 nb\_ventes int;  
begin  
 select idlocalisation into id\_localisation from localisation where code\_postal = codepostal;  
 select count(\*) into nb\_ventes from services where idlocalisation = id\_localisation;  
 dbms\_output.*put\_line*('Il y a ' || nb\_ventes || ' proposition(s) de services.');  
EXCEPTION  
 WHEN no\_data\_found THEN DBMS\_OUTPUT.*PUT\_LINE*('Code postal inexistant !');  
 WHEN others THEN DBMS\_OUTPUT.*PUT\_LINE*('Erreur !');  
end;  
/  
  
BEGIN  
 *affiche\_services\_localisation*('57340');  
end;  
/

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Procédure affichant le nombre de ventes pour une date donnée  
 \*/  
create or replace procedure *affiche\_vente\_toutes\_villes*(dateh planning.date\_heure%type)  
 is  
 nb\_ventes int;  
begin  
 select count(\*) into nb\_ventes from planning where dateh = date\_heure;  
 dbms\_output.*put\_line*('Il y a ' || nb\_ventes || ' création(s) de vente le ' || dateh);  
EXCEPTION  
 WHEN no\_data\_found THEN DBMS\_OUTPUT.*PUT\_LINE*('Date inexistante !');  
 WHEN others THEN DBMS\_OUTPUT.*PUT\_LINE*('Erreur !');  
end;  
/  
  
BEGIN  
 *affiche\_vente\_toutes\_villes*(TO\_DATE('27/05/22', 'dd/mm/yy'));  
end;  
/

—------------------------------------------------------------------------------------------------------------------------

Indicateur 2 :

/\*\*  
 Procédure affichant toutes les ventes créés/effectuées pour toutes les dates  
 \*/  
create or replace procedure *affiche\_vente\_tout\_planning* is  
 nb\_ventes int;  
 dateh planning.date\_heure%type;  
begin  
 for i in (select \* from bien)  
 loop  
 select count(\*) into nb\_ventes from bien where i.IDPLANNING = idplanning;  
 select date\_heure into dateh from planning where i.IDPLANNING = idplanning;  
 dbms\_output.*put\_line*('Il y a ' || nb\_ventes || ' vente(s) le ' || dateh);  
 end loop;  
end;  
/  
  
BEGIN  
 *affiche\_vente\_tout\_planning*();  
end;  
/