

Requête utile

Liste de requête utiles pour un SGBD Oracle

```
-- Requetes Utile

-- Modification Mot de passe utilisateur
-- On modifie l'utilisateur WIKI_PROD et on remplace le mdp DORP_IKIW par IKIW
ALTER USER WIKI_PROD IDENTIFIED BY IKIW REPLACE DORP_IKIW;
-- Inversement
ALTER USER WIKI_PROD IDENTIFIED BY DORP_IKIW REPLACE IKIW;

-- Débloquer le compte (se bloque au bout de 10 tentatives)
ALTER USER WIKI_PROD ACCOUNT UNLOCK;

-- Récupérer toutes les contraintes existantes pour une table
SELECT * FROM USER_CONSTRAINTS WHERE TABLE_NAME = '<tablename>';

-- Inner join avec COUNT(*)
0 != (SELECT COUNT(*) FROM TABLE2 WHERE TABLE2.ID_USER = USER.ID)

-- Hiérarchie des Entités
SELECT LPAD(ENTITY.TITRE, LENGTH(ENTITY.TITRE) + (LEVEL - 1) * 3, '+') AS Organigramme,
ENTITY.ID, ENTITY.CODE, ENTITY.ID_ENTITY_FATHER, LEVEL FROM ENTITY
CONNECT BY ENTITY.ID = PRIOR ENTITY.ID_ENTITY_FATHER
START WITH ENTITY.ID = 1
ORDER BY LEVEL DESC;

-- Récupération des informations de la BDD
SELECT
[]col.table_name,
[]col.column_name,
[]col.data_type,
```

```

col.data_length
FROM
sys.all_tab_columns col
INNER JOIN sys.all_tables t ON
col.owner = t.owner
AND col.table_name = t.table_name
WHERE
col.owner = '<schemaname>'
AND col.table_name = '<tablename>' -- optionnel
ORDER BY
col.table_name;

-- Parcours d'un XML avec condition
-- Si on effectue le traitement sur une colonne de type XMLTYPE le resultat pourra s'auto
formater en XML (ne pas oublier d'enlever GetClobVal)
-- Ici XML_COLUMN est la colonne au format XML qui est dans TABLE1.
SELECT id, code,
XMLQuery(
  ' for $i in /Root/Options/Option
  where $i/Item[@code="MON_CODE_ITEM"]
  return $i/Item/Value/@value ' PASSING Xmltype(XML_COLUMN) RETURNING CONTENT).GetClobVal()
AS res
FROM TABLE1 WHERE ID IN (SELECT ID FROM TABLE2 WHERE CODE = 'DEMO' AND TYPE = '1');

-- Procedure pour retourner le résultat d'une requête
-- Avec 2 requêtes
DECLARE
value NVARCHAR2(2000);
code NVARCHAR2(255);
c1 SYS_REFCURSOR;
CURSOR cursorData IS SELECT CODE FROM TABLE1 WHERE CATEGORY = 'MY_CATEGORY' AND TEMPLATE =
'EXPORT';
BEGIN
OPEN cursorData;
LOOP
FETCH cursorData
INTO code;
EXIT

```

```

    WHEN cursorData%NOTFOUND;
    value := 'CODE LIKE '%" || code || '%" OR ' || value;
END LOOP;
value := SUBSTR(value,1,LENGTH(value)-3);
CLOSE cursorData;

```

```

open c1 for 'SELECT ID, CODE, TITRE
    FROM TABLE2 WHERE
        TEMPLATE IN (SELECT CODE FROM TABLE3 WHERE ' || value ||')
        AND CATEGORY = 'MY_CATEGORY' AND TYPE = '1';
-- Retourne le résultat de la 2eme requête
DBMS_SQL.RETURN_RESULT(c1);
END;
/

```

```

-- Procedure pour parcourir le résultat des 2 requêtes
-- Dans ce cas la procédure parcourt toutes les tables contenant la colonne "CODE_RESOURCE"
DECLARE
    value NVARCHAR2(255);
    res NVARCHAR2(255);
    c1 SYS_REFCURSOR;
    CURSOR cursorData IS SELECT TABLE_NAME FROM USER_TAB_COLUMNS WHERE COLUMN_NAME =
'CODE_RESOURCE';
BEGIN
OPEN cursorData;
LOOP FETCH cursorData INTO value;
EXIT WHEN cursorData%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(value);
open c1 for 'SELECT COUNT(*) FROM ' || value ||' WHERE CODE_RESOURCE = 'MY_CODE'';
LOOP FETCH c1 INTO res;
EXIT WHEN c1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(res ||' ==> ' || value);
END LOOP;
CLOSE c1;
END LOOP;
CLOSE cursorData;
END;

```

```

-- Parcourir tous les schémas
DECLARE

```

```

value NVARCHAR2(255);
res NVARCHAR2(500);
c1 SYS_REFCURSOR;
CURSOR cursorData IS SELECT USERNAME FROM dba_users WHERE DEFAULT_TABLESPACE = 'USERS' AND
ACCOUNT_STATUS = 'OPEN' AND USERNAME NOT IN ('WIKI_TEST', 'DEMO', 'TEST');
BEGIN
OPEN cursorData;
LOOP FETCH cursorData INTO value;
EXIT WHEN cursorData%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(value);
DBMS_OUTPUT.PUT_LINE('----- ' || value || ' -----');
open c1 for 'SELECT URL_IMAGE FROM ' || value || '.UPLOAD WHERE ID IN (SELECT ID_IMAGE FROM
|| value || '.FOLDER WHERE ID IN (2,3,4))';
LOOP FETCH c1 INTO res;
EXIT WHEN c1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE( value || ' ==> ' || res);
END LOOP;
CLOSE c1;
END LOOP;
CLOSE cursorData;
END;

-- Récupère l'historique des requêtes avec leurs temps d'exécution et la date d'exécution
select LAST_LOAD_TIME, ELAPSED_TIME, MODULE, SQL_TEXT elapsed from v$sql order by
LAST_LOAD_TIME DESC;

-- ACTIVER/DESACTIVER UN TRIGGER
ALTER TRIGGER update_history DISABLE;
ALTER TRIGGER update_history ENABLE;

```

Revision #3

Created 2 February 2023 16:45:27 by gpatruno

Updated 2 February 2023 17:10:23 by gpatruno