Requêtes Outils DSI 2.0

Le ***user_id*** correspond à la première lettre du prénom et le nom de famille
(Ex: jdupont -> Jean DUPONT)

Liste des BI

```
SELECT
    b.BIDTP_NUM AS NOM_BI,
    CASE
        WHEN b.BIODO_PRG IS NULL THEN 'Non renseigné'
        WHEN b.BIODO_PRG = 'WORDUNG'  THEN 'Document Word (.doc)'
        WHEN b.BIODO_PRG = 'EXCELUNG' THEN 'Document Excel (.xls)'
        WHEN b.BIODO_PRG = 'XLSXUNG'  THEN 'Document Excel (.xlsx)'
        WHEN b.BIODO_PRG = 'WRDNGPDF' THEN 'Document PDF (.pdf)'
    ELSE 'Type inconnu'
    END AS DOCUMENT_TYPE,
    b.BIDTP_LIB AS DESCRIPTION_BI,
    b.BIDTP_NOM AS NOM_DOCUMENT,
    CASE
        WHEN i.ICTRS_DSC IS NULL THEN 'Pas de description'
        ELSE DBMS_LOB.SUBSTR(i.ICTRS_DSC, 4100)
    END AS DESCRIPTION_PLUS
FROM
    BIDTP b,
    ICTRS i
WHERE
    i.ICTRS_NOM = b.BIDTP_NUM
```

Mot de passe des utilisateurs

```
SELECT
    motdepasse(mguti_cod) AS mdp,
    CASE MGUTI_TEMMDP
        WHEN 0 THEN 'Mdp valide'
        WHEN 1 THEN 'A changer à la prochaine connexion'
        WHEN 2 THEN 'Mdp verrouilé'
    END AS Statut,
    CASE
        WHEN MGUTI_MOTPEXP < sysdate THEN ' - Mot de passe expiré'
    END AS expiration
FROM
    mguti
WHERE
    mguti_cod = upper('user_id')
```

Débloquer mot de passe des utilisateurs

```
Update MGUTI set MGUTI_TEMMDP = 0 where MGUTI_COD='user_id'
```

Réinitialisé le mot de passe des utilisateurs.

```
Update MGUTI set MGUTI_TEMMDP = 1, MGUTI_MOTP='ZE19' where
MGUTI_COD='user_id'
-- Mot de passe à changer à la prochaine connexion
```

Liste des utilisateurs Actif (MGUTI_COD = 'A')

```
SELECT
    TOTIE_COD AS NUM_TIERS,
    MGUTI_COD AS CODE_UTILISATEUR,
    MGGUT_COD AS GROUPE,
    MGUTI_NOM AS NOM,
    MGUTI_PRENOM AS PRENOM,
    MGUTI_ETA AS ETAT,
    MGGUT_CODWEB AS CODE_WEB,
    MGMWB_COD AS CODE_ULIS,
    MGUTI_DERCON AS DERNIERE_CONNEXION
FROM
    MGUTI
WHERE
    MGUTI_ETA = 'A'
    --AND MGUTI_COD = 'USER_ID' --Pour rechercher un utilisateur
ORDER BY
    TOTIE_COD
```

Connexion Outils DSI (sans SSO)

```
SELECT MGUTI_COD, motdepasse(mguti_cod) FROM MGUTI WHERE MGUTI_COD =
'USER_ID'
```

Onglet Locataire

Recherche par ESI

```
SELECT
  F_Libelle_Occupant(a.paesi_num, SYSDATE) AS nom_loc,
  a.paesi_codext                           AS esi,
  NVL(
    get_signataire( GET_CONTRAT_DT(a.paesi_num, SYSDATE) ),
```

```
    (SELECT totie_cod
       FROM glocc
      WHERE glcon_num = GET_CONTRAT_DT(a.paesi_num, SYSDATE)
        AND ROWNUM = 1)
  ) AS tiers,
  F_CONTRAT_ESI(a.paesi_num, SYSDATE) AS contrat
FROM syn_pat a
WHERE a.paesi_codext LIKE REPLACE('%PI0106.01.05.0027%', '*', '%')
  AND F_Libelle_Occupant(a.paesi_num, SYSDATE) IS NOT NULL
  AND F_CONTRAT_ESI(a.paesi_num, SYSDATE) IS NOT NULL
ORDER BY 1;
```

Recherche par Contrat

Recherche par Intitule

```
SELECT
    DISTINCT cactr_num
FROM
    cactr
WHERE
    CACTR_DTF IS NULL
    AND CACTR_TEMP = 'T'
    AND caint_num = 12356
```

Récupérer la dette

```
select get_dette({0}) from dual
```

Info locataire à afficher (onglet locataire)

```
SELECT
    MGQUA_COD || ' ' || TODPP_PRE || ' ' || TODPP_NOM AS Nom,
    mef_no_tel(TODPP_NUMPOR) AS TODPP_NUMPOR,
    a.totie_cod AS NoTiers
FROM
    todpp a,
    glcsi b
WHERE
    b.totie_cod = a.totie_cod
    AND glcon_num = 18181
```

Requete (onglet Extraction)

```
SELECT
    a.PAESI_CODEXT AS ESI
FROM
    PAESI a
    JOIN SYN_PAT x
        ON x.paesi_num = a.paesi_num
WHERE
    -- Selon la valeur de TBGroupe.Text :
    -- Cas PCH :
    -- a.panes_cod NOT IN ('GPE', 'BAT', 'ESC')
    -- AND a.PAESI_DATSOR IS NULL
    -- ORDER BY a.PAESI_CODEXT
    --
    -- Cas EST :
    -- UPPER(x.agence) LIKE '%EST'
    -- AND a.panes_cod NOT IN ('GPE', 'BAT', 'ESC')
    -- AND a.PAESI_DATSOR IS NULL
    --
    -- Cas autres groupes prédéfinis :
    -- UPPER(x.agence) LIKE '%<TBGroupe>'
    -- AND a.panes_cod NOT IN ('GPE', 'BAT', 'ESC')
    -- AND a.PAESI_DATSOR IS NULL
    --
    -- Cas par défaut :
    -- x.groupe = '<TBGroupe>'
    -- AND a.panes_cod NOT IN ('GPE', 'BAT', 'ESC')
    -- AND a.PAESI_DATSOR IS NULL
```

Ensuite, **selon les cases cochées** :

- Des colonnes supplémentaires s'ajoutent à la clause `SELECT`
- Des `JOIN` supplémentaires s'ajoutent à la clause `FROM`
- Des sous-requêtes peuvent être ajoutées pour aller chercher d'autres informations (ex. `Adresse`, `Typologie`, `Locataire`, etc.)

📌 Exemple avec **toutes** les options cochées (pour visualiser le maximum de colonnes) :

```
SELECT
    a.PAESI_CODEXT AS ESI,
    b.PAETG_COD AS Etage,
    b.PAETY_COD AS Typologie,
    a.PANES_COD AS Nature,
    md.mgadr_numvoi || ' ' || md.mgtvo_cod || ' ' || md.mgadr_nomvoi || '
' ||
        md.mgadr_libco1 || ' ' || md.mgadr_libco2 || ' ' ||
md.mgadr_libco3 AS Adresse,
    md.mgadr_codpos AS CP,
    md.mgadr_libcom AS Commune,
```

```
    x.groupe AS Groupe,
    GETAGENCEESI(a.PAESI_NUM, 'L') AS Agence,
    (SELECT PAESU_LIB
     FROM paesu
     WHERE paesu_cod = (SELECT paesu_cod
                        FROM palst
                        WHERE paesi_num = a.paesi_num
                          AND PALST_DTF IS NULL)
    ) AS Statut_ESI,
    FGETSURFACE(a.paesi_num, 'SH') AS Surface_Habitable,
    (SELECT MAX(PAETM_VAL)
     FROM PAETM
     WHERE PATAN_COD = 'SU'
       AND PAESI_NUM = a.PAESI_NUM
    ) AS Surface_utile,
    (SELECT PAEST_VAL
     FROM PAEST
     WHERE paesi_num = a.paesi_num
       AND MGZDE_COD = 'NODRE'
       AND MGENT_COD = 'PAESI'
    ) AS Numero_DRE,
    GET_FINANCEMENT(a.paesi_num) AS Cat_Financement,
    FGETRESERVATAIRE(a.paesi_num) AS Réservataire,
    c.GLCON_NUM AS Contrat,
    c.GLCON_NUMVER AS Version_Contrat,
    d.CAINT_NUM AS Intitule,
    e.Totie_COD AS Tiers,
    f.GLREC_TITRE || ' ' || f.GLREC_INTCOU1 AS Libelle_Intitule,
    GET_PORTABLE(e.Totie_COD) AS Tel_Portable,
    GET_AGE(e.Totie_COD) AS Age,
    GET_ASSURANCE_A_JOUR(c.GLCON_NUM) AS Assurance_A_jour,
    DECODE(GET_PRELEVE(d.caint_num), 0, 'NON', 'OUI') AS Prelevement,
    (SELECT TO_CHAR(GLCON_DTD, 'dd/mm/yyyy')
     FROM GLCON
     WHERE GLCON_NUM = c.GLCON_NUM
       AND GLCON_NUMVER = c.GLCON_NUMVER
    ) AS Date_Debut_Contrat,
    (SELECT TO_CHAR(GLCON_DTF, 'dd/mm/yyyy')
     FROM GLCON
     WHERE GLCON_NUM = c.GLCON_NUM
       AND GLCON_NUMVER = c.GLCON_NUMVER
    ) AS Date_Fin_Contrat,
    (SELECT TRIM(mgadr.mgadr_numvoi || ' ' || mgadr.mgtvo_cod || ' ' ||
                 mgadr.mgadr_nomvoi || ' ' || mgadr.mgadr_libco1)
     FROM mgadr
     WHERE mgadr_num = f.MGADR_NUM
    ) AS Adresse_Regroupement,
    (SELECT TRIM(MGADR.MGADR_CODPOS)
     FROM mgadr
     WHERE mgadr_num = f.MGADR_NUM
```

```
    ) AS CP_Regroupement,
    (SELECT TRIM(MGADR.MGADR_LIBCOM)
     FROM mgadr
     WHERE mgadr_num = f.MGADR_NUM
    ) AS Ville_Regroupement,
    (SELECT GLCZP_VAL
     FROM GLCZP
     WHERE GLCON_NUM = c.GLCON_NUM
       AND GLCON_NUMVER = c.GLCON_NUMVER
       AND MGZDE_COD = 'IMMAT'
       AND ROWNUM = 1
    ) AS immatriculation,
    (SELECT GLCZP_VAL
     FROM GLCZP
     WHERE GLCON_NUM = c.GLCON_NUM
       AND GLCON_NUMVER = c.GLCON_NUMVER
       AND MGZDE_COD = 'MARQ'
       AND ROWNUM = 1
    ) AS Marque_vehicule,
    F_GET_NBOCC(c.GLCON_NUM, c.GLCON_NUMVER) AS NB_Occupants,
    F_GET_NBOCC(c.GLCON_NUM, c.GLCON_NUMVER) - GET_NB_ENFANTS(c.GLCON_NUM)
AS NB_ADULTES,
    GET_NB_ENFANTS(c.GLCON_NUM) AS NB_Enfants,
    GET_RI(c.GLCON_NUM, c.GLCON_NUMVER) AS Revenu_Imposable_Menage,
    GET_LOYER(c.GLCON_NUM, c.GLCON_NUMVER) AS Loyer_Contrat,
    GET_CHARGES(c.GLCON_NUM, c.GLCON_NUMVER) AS Charges_Contrat,
    GET_DETTE(d.CAINT_NUM) AS Dette_Intitule,
    GET_APL(c.GLCON_NUM, c.GLCON_NUMVER) AS APL,
    GET_RLS(c.GLCON_NUM, c.GLCON_NUMVER) AS RLS,
    GET_LOYERMAX_CONV(a.paesi_num) AS Loyer_Plafond
FROM
    PAESI a
    JOIN SYN_PAT x
        ON x.paesi_num = a.paesi_num
    JOIN PAIFG b
        ON b.paesi_num = a.paesi_num
    JOIN PAEAD p
        ON p.paesi_num = a.paesi_num
    JOIN MGADR md
        ON md.mgadr_num = p.mgadr_num
    LEFT JOIN GLELC c
        ON c.paesi_num = a.paesi_num
       AND (SELECT GLCON_TEMCA FROM GLCON
            WHERE glcon_num = c.glcon_num
              AND glcon_numver = c.glcon_numver) = 'F'
       AND c.GLELC_DTF IS NULL
    LEFT JOIN CACTR d
        ON d.CACTR_NUM = c.GLCON_NUM
       AND d.CACTR_VRS = c.GLCON_NUMVER
    LEFT JOIN GLREC f
```

```
        ON f.CAINT_NUM = d.CAINT_NUM
    LEFT JOIN CAICL e
        ON e.caint_num = d.caint_num
       AND e.CAICL_TEMP = 'T'
       AND e.CAICL_DTF IS NULL
WHERE
    a.panes_cod NOT IN ('GPE', 'BAT', 'ESC')
    AND a.PAESI_DATSOR IS NULL
ORDER BY a.PAESI_CODEXT;
```

Onglet Engagement

Affichage des données de l'engagement (bouton Check)

```
--Type d'engagement (TATEN_COD)
SELECT TATEN_COD FROM TAENG WHERE ICEXE_NUM = ? AND TAENG_NUM = ? AND
TOTIE_CODSCTE = ?

-- ESO administratif (TOESO_COD)
SELECT TOESO_COD FROM TAENG WHERE ICEXE_NUM = ? AND TAENG_NUM = ? AND
TOTIE_CODSCTE = ?

-- Responsable (TOTIE_COD)
SELECT TOTIE_COD FROM TAENR WHERE ICEXE_NUM = ? AND TAENG_NUM = ? AND
TOTIE_CODSCTE = ?

-- Marché rattaché (TAMAC_NUM)
SELECT TAMAC_NUM FROM TAENG WHERE ICEXE_NUM = ? AND TAENG_NUM = ? AND
TOTIE_CODSCTE = ?

-- Lot rattaché (TAMAL_REF) - Jointure complexe
SELECT TAMAL_REF FROM TAMAL a, TAENG b WHERE ICEXE_NUM = ? AND TAENG_NUM =
? AND TOTIE_CODSCTE = ? AND b.tamac_num = a.tamac_num AND b.tamal_cod =
a.tamal_cod AND tavma_numver = (SELECT MAX(tavma_numver) FROM tamal WHERE
tamac_num = a.tamac_num AND tamal_cod = a.tamal_cod)

-- Vérification pluriannuel
SELECT CASE WHEN TAENG_TEMPLURI = 'F' THEN 0 ELSE 1 END FROM TAENG WHERE
ICEXE_NUM = ? AND TAENG_NUM = ?
```

Mise à jour des données de l'engagement

```
-- Pluriannuel Oui/NON

UPDATE TAENG SET TAENG_TEMPLURI = '{T/F}' WHERE ICEXE_NUM =
{TBexercice.Text} AND TAENG_NUM = {tbnumeng.Text}
```

```
AND TOTIE_CODSCTE = '{DDLSoc.SelectedValue}'

-- Mise à jour du type engagement

UPDATE TAENG SET TATEN_COD = '{TBtypeeng.Text}' WHERE ICEXE_NUM =
{TBexercice.Text} AND TAENG_NUM = {tbnumeng.Text}
AND TOTIE_CODSCTE = '{DDLSoc.SelectedValue}'

-- Mise àjour du Status
--UPDATE TAFAC SET TASTA_COD = '{DDLStatut.SelectedValue}' WHERE TAFAC_NUM
= {TBNoFac.Text}
--AND ICEXE_NUM = {DDLexercice.SelectedValue}

-- Mise à jour de l'ESO
UPDATE TAENG SET TOESO_COD = '{tbesoadm.Text}' WHERE ICEXE_NUM =
{TBexercice.Text} AND TAENG_NUM = {tbnumeng.Text}
AND TOTIE_CODSCTE = '{DDLSoc.SelectedValue}'

-- Modification du responsable engagement

DELETE FROM TAENR WHERE ICEXE_NUM = {TBexercice.Text} AND TAENG_NUM =
{tbnumeng.Text}
AND TOTIE_CODSCTE = '{DDLSoc.SelectedValue}'

INSERT INTO TAENR
(TOTIE_CODSCTE, ICEXE_NUM, TAENG_NUM, TAITL_COD, TOTIE_COD, TAENR_DATCRE)
VALUES
({DDLSoc.SelectedValue}, {TBexercice.Text}, '{tbnumeng.Text}',
'RESEN', '{tbrespeng.Text}', TRUNC(SYSDATE))
```

Admin

```
-- 1) Table ADM_ADMIN
CREATE TABLE ADM_ADMIN (
    ID_UTILISATEUR VARCHAR2(50) NOT NULL,
    IS_ADMIN CHAR(1) DEFAULT 'N' NOT NULL, -- 'Y' ou 'N'
    DATE_ADMIN DATE DEFAULT SYSDATE NOT NULL,
    CONSTRAINT PK_ADM_ADMIN PRIMARY KEY (ID_UTILISATEUR),
    CONSTRAINT CK_ADM_ADMIN_IS_ADMIN CHECK (IS_ADMIN IN ('Y','N'))
);

-- 2) Table ADM_PAGE_ACCESS
CREATE TABLE ADM_PAGE_ACCESS (
    ID_UTILISATEUR VARCHAR2(50) NOT NULL,
    CODE_PAGE VARCHAR2(64) NOT NULL, -- ex: route symfony
'admin_dashboard'
    NOM_PAGE VARCHAR2(128) NOT NULL, -- label affiché 'Dashboard'
    DATE_ADMIN DATE DEFAULT SYSDATE NOT NULL,
```

```
    CONSTRAINT PK_ADM_PAGE_ACCESS PRIMARY KEY (ID_UTILISATEUR, CODE_PAGE),
    CONSTRAINT FK_APA_USER FOREIGN KEY (ID_UTILISATEUR)
    REFERENCES ADM_ADMIN (ID_UTILISATEUR)
ON DELETE CASCADE
);

-- 3) Index utile (recherche par page)
CREATE INDEX IX_APA_CODE_PAGE ON ADM_PAGE_ACCESS (CODE_PAGE);
```

```
-- Déclarer un utilisateur comme admin
MERGE INTO ADM_ADMIN a
USING (SELECT 'PCH' AS ID_UTILISATEUR FROM DUAL) s
ON (a.ID_UTILISATEUR = s.ID_UTILISATEUR)
WHEN MATCHED THEN UPDATE SET a.IS_ADMIN = 'Y', a.DATE_ADMIN = SYSDATE
WHEN NOT MATCHED THEN INSERT (ID_UTILISATEUR, IS_ADMIN, DATE_ADMIN)
VALUES ('PCH', 'Y', SYSDATE);

-- Donner des accès à PCH (adapter selon vos routes/labels)
MERGE INTO ADM_PAGE_ACCESS t
USING (SELECT 'PCH' ID_UTILISATEUR, 'admin_dashboard' CODE_PAGE,
'Dashboard' NOM_PAGE FROM DUAL UNION ALL
SELECT 'PCH', 'admin_engagement', 'Engagements' FROM DUAL UNION ALL
SELECT 'PCH', 'admin_extraction', 'Extraction' FROM DUAL UNION ALL
SELECT 'PCH', 'admin_edition_bureautique', 'Éditions bureautiques' FROM
DUAL
) s
ON (t.ID_UTILISATEUR = s.ID_UTILISATEUR AND t.CODE_PAGE = s.CODE_PAGE)
WHEN MATCHED THEN UPDATE SET t.NOM_PAGE = s.NOM_PAGE, t.DATE_ADMIN =
SYSDATE
WHEN NOT MATCHED THEN INSERT (ID_UTILISATEUR, CODE_PAGE, NOM_PAGE,
DATE_ADMIN)
VALUES (s.ID_UTILISATEUR, s.CODE_PAGE, s.NOM_PAGE, SYSDATE);
```

```
--Tables d'accès des pages
SELECT * FROM ADM_PAGE_ACCESS

-- Tables des utilisateurs Admin
SELECT * FROM ADM_ADMIN

--Jointure des 2 tables
SELECT * FROM ADM_PAGE_ACCESS, ADM_ADMIN WHERE
ADM_PAGE_ACCESS.ID_UTILISATEUR = ADM_ADMIN.ID_UTILISATEUR
```

```
--Pour rendre un utilisateur admin tout de suite
MERGE INTO ADM_ADMIN a
USING (SELECT 'VOTRE_ID' AS ID_UTILISATEUR FROM DUAL) s
ON (a.ID_UTILISATEUR = s.ID_UTILISATEUR)
```

```
WHEN MATCHED THEN UPDATE SET a.IS_ADMIN = 'Y', a.DATE_ADMIN = SYSDATE
WHEN NOT MATCHED THEN INSERT (ID_UTILISATEUR, IS_ADMIN, DATE_ADMIN)
VALUES ('VOTRE_ID', 'Y', SYSDATE);

--Accorder un page
MERGE INTO ADM_PAGE_ACCESS t
USING (SELECT 'VOTRE_ID' ID_UTILISATEUR, 'admin_user_access' CODE_PAGE,
'Administration' NOM_PAGE FROM DUAL) s
ON (t.ID_UTILISATEUR = s.ID_UTILISATEUR AND t.CODE_PAGE = s.CODE_PAGE)
WHEN MATCHED THEN UPDATE SET t.NOM_PAGE = s.NOM_PAGE, t.DATE_ADMIN =
SYSDATE
WHEN NOT MATCHED THEN INSERT (ID_UTILISATEUR, CODE_PAGE, NOM_PAGE,
DATE_ADMIN)
VALUES (s.ID_UTILISATEUR, s.CODE_PAGE, s.NOM_PAGE, SYSDATE);
```

Proposition (Suppression)

```
-- Tables des candidats
SELECT TODPP_PRE AS PRENOM,
    TODPP_NOM AS NOM,
    TODPP_DATNAI AS DATE_NAISSANCE,
    ACPRS_NUM AS NUMERO_PROPOSITION,
    TOTIE_COD AS NUMERO_TIERS,
    ACDOS_NUM AS NUMERO_DOSSIER,
    ACPRC_DATCRE AS DATE_CREATION
FROM ACPRC
    WHERE ACPRS_NUM = :num

-- Tables des propositions
SELECT
    ACPRS_NUM,
    TOTIE_COD_RES,
    TOTIE_LIB,
    PAESI_CODEXT,
    PANES_COD,
    MGADR_LIB,
    MGADR_CODPOS,
    MGADR_LIBCOM,
    TOTIE_LIB_GAR,
    ACPRS_DATCRE
FROM ACPRS
    WHERE ACPRS_NUM = :num

-- Suppression de tous les candidats d'une proposition
DELETE FROM ACPRC WHERE ACPRS_NUM = :num

-- Suppression d'un candidats
DELETE FROM ACPRC
```

```
WHERE ACPRS_NUM = :num
AND TOTIE_COD = :tiers
AND ACDOS_NUM = :dossier

-- Suppression Propositions
DELETE FROM ACPRS WHERE ACPRS_NUM = :num
```

Utilisateur BECKREL

```
--Liste des utilisateur Beckrel
SELECT EMAIL, PHONE FROM BECKREL_V_USERS ORDER BY EMAIL

--Verifie si le tiers existe dans TOZD2
SELECT COUNT(*) FROM TOZD2 WHERE TOTIE_COD = :tiers

-- Remplit TOZD2_VALPHA avec l'email
UPDATE TOZD2
SET TOZD2_VALPHA = :email
WHERE TOTIE_COD = :tiers

--Insertion dans la table BECKREL_USER_ACCESS
INSERT INTO
    BECKREL_USERS_ACCESS (TOTIE_COD)
VALUES
    (:tiers)
```

Utilisateur SOWELL

```
-- Liste des utilisateur Sowell
SELECT
    FIRST_NAME, LAST_NAME, CODE, EMAIL
FROM SOWELL_V_USER
ORDER BY LAST_NAME, FIRST_NAME

-- Recherche d'un utilisateur (non utilisé)
SELECT
    FIRST_NAME, LAST_NAME, CODE, EMAIL
FROM SOWELL_V_USER
WHERE UPPER(EMAIL) = UPPER(:email)

-- Ajout d'un utilisateur pour l'application Sowell
UPDATE TOZD2
SET TOZD2_VALPHA = :email
WHERE TOTIE_COD = :code
```

Logement

Mode OP