

Designer et l'ingénierie du logiciel

Les données calculées par les applications Web PL/SQL

P.-A. Sunier, ISNet Neuchâtel avec le concours de C. Kohler et P. Ferrara

1	Introduction.....	1
2	Besoin de données calculées	2
3	Champs calculés.....	4
3.1	Spécification.....	4
3.2	Modalités de calcul	4
3.2.1	Calcul par dérivation	5
3.2.2	Calcul par une fonction PL/SQL.....	5
3.2.3	Formulaire.....	6
4	Table d'agrégat	7
5	Conclusions.....	8
6	Liens utiles	8

1 Introduction

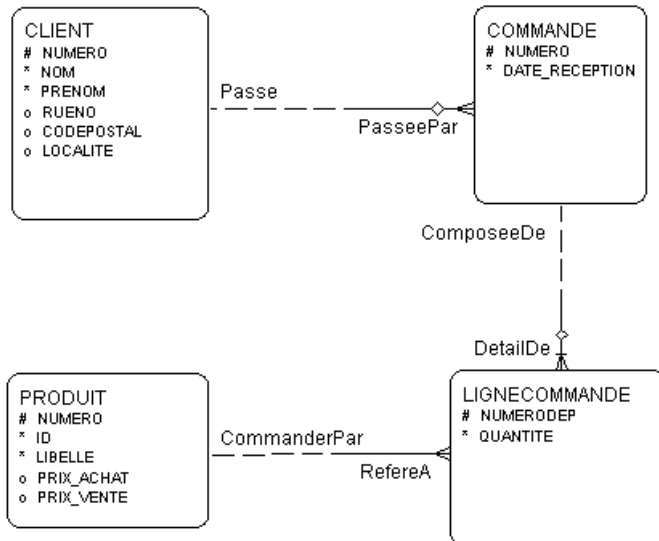
Dans le premier article consacré à *Designer*, **Newsletter 2/2003 d'avril 2003**, sous le titre **Notions élémentaires**, nous avons présenté les concepts de *Designer* sur la base d'une approche d'ingénierie du logiciel alliant méthodologie, normalisation et standardisation, qualité et réutilisation. Nous avons décrit succinctement les modules de traitement et le générateur d'application Web PL/SQL.

Dans le deuxième article, **Newsletter 3/2003 d'août 2003**, sous le titre **Couplage entre structure de données et modules de traitement**, nous avons présenté les structures essentielles de manipulation de données de base.

Dans cet article, nous expliquerons comment fournir à l'utilisateur des données issues non plus des tables manipulées par les modules mais résultant de traitements plus ou moins sophistiqués. Ces données, que nous appelons « calculées », peuvent être produites de deux manières distinctes : par les modules de traitement eux-mêmes ou par le serveur de données via la APIS de tables.

Nous présenterons ci-après les mécanismes d'alimentation de données calculées par les modules de traitement; dans un prochain article, nous décrirons les mécanismes d'alimentation de données calculées fournis par les APIs de tables.

2 Besoin de données calculées



Dans le deuxième article, **Couplage entre structure de données et modules de traitement**, nous avons présenté le modèle conceptuel de données reproduit en figure 1.

Figure 1 - MCD des commandes

Après transformation de ce modèle conceptuel de données en un modèle logique de données, *Server Model*, nous avons créé le module de traitement des commandes reproduit en figure 2.

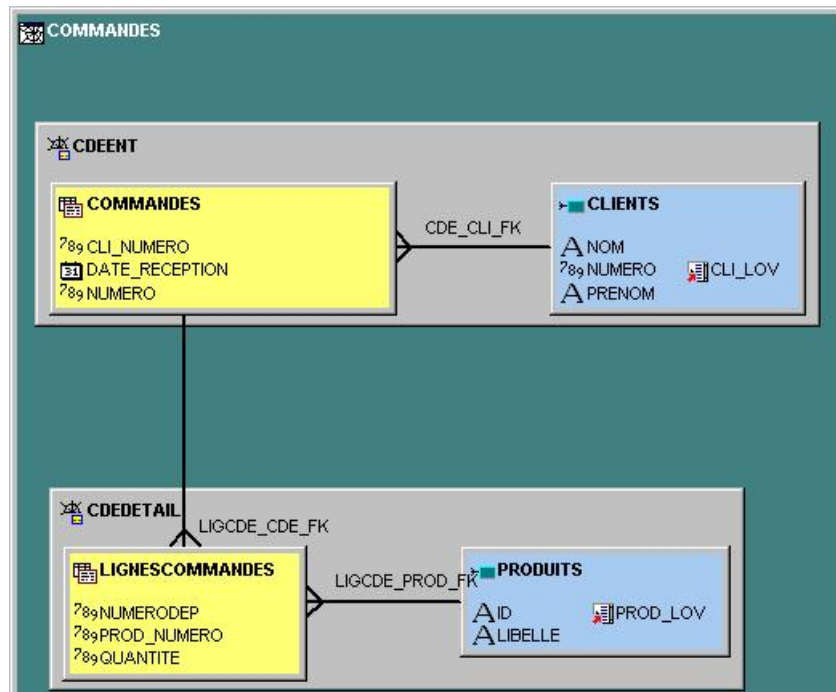


Figure 2 - Module des commandes

Après la génération par *Web PL/SQL* de ce module, l'utilisateur dispose d'un formulaire (figure 3) qui lui permet de traiter une commande.

Composant d'entête de commandes

Date de réception: 11.01.2000 [CAL](#)

Numéro (client): 2

Nom (client): Dubois

Prénom (client): Anne

Mettre à jour Supprimer Rétablir Nouveau

Détail de la commande

Id	Libelle	Quantite
10.30	Moyeu avant	2
10.32	Patins de freins	4
12.44	Dérailleur	1

Nouveau

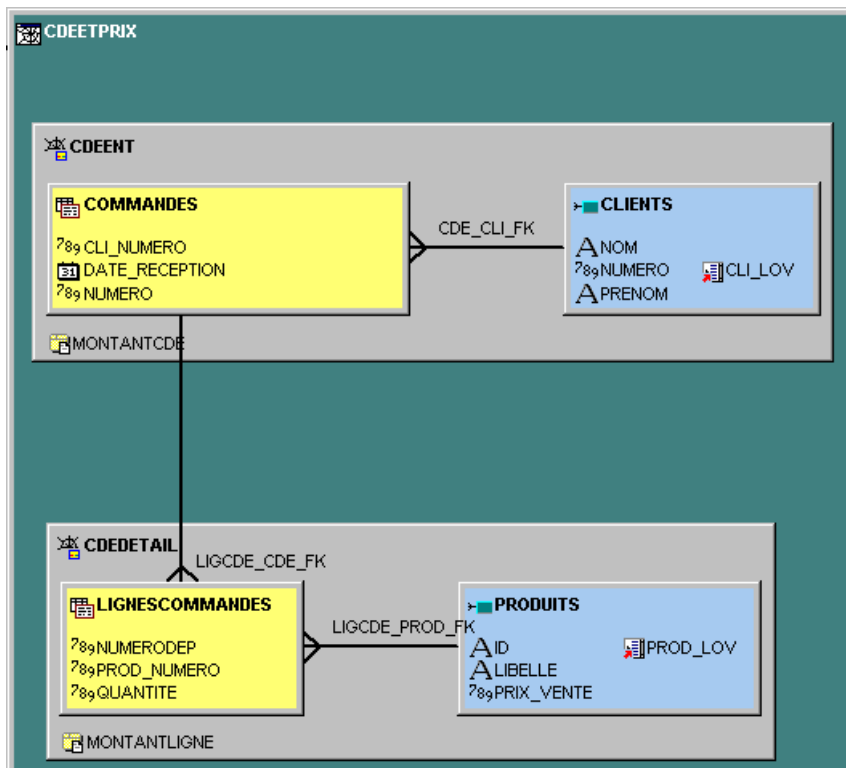
Figure 3 - Formulaire des commandes

Ce formulaire fournit à l'utilisateur les données de base d'une commande ; mais, en plus de ces données de base, il serait judicieux de fournir à l'utilisateur des informations plus pertinentes telles que le montant d'une ligne de commande et le montant total d'une commande.

Ces deux informations seront fournies à l'utilisateur sous forme de données calculées ; et, comme nous l'avons indiqué en introduction, nous effectuerons ce calcul au niveau du module de traitement à l'aide de champs calculés.

3 Champs calculés

3.1 Spécification



La figure 4 représente un module de traitement des commandes comprenant deux champs calculés : **MONTANTCDE** et **MONTANTLIGNE**. Un champ calculé est rattaché à un composant de module sans être associé à une colonne de table.

MONTANTCDE est associé au composant maître et devra afficher le montant total de la commande. **MONTANTLIGNE** est associé au composant de détail et devra afficher le montant d'une ligne de commande.

Figure 4 - Module des commandes avec champs calculés

3.2 Modalités de calcul

Designer offre plusieurs méthodes d'alimentation d'un champ calculé.

La figure 5 illustre le paramétrage de la méthode d'alimentation du champ calculé **MONTANTCDE** ; ce champ est alimenté par une fonction PL/SQL exécutée sur le serveur de base de données. Nous verrons, par la suite, l'alimentation du montant de ligne, **MONTANTLIGNE**, par une expression SQL et le calcul du montant de commande par une fonction d'agrégat comme alternative à la fonction PL/SQL.

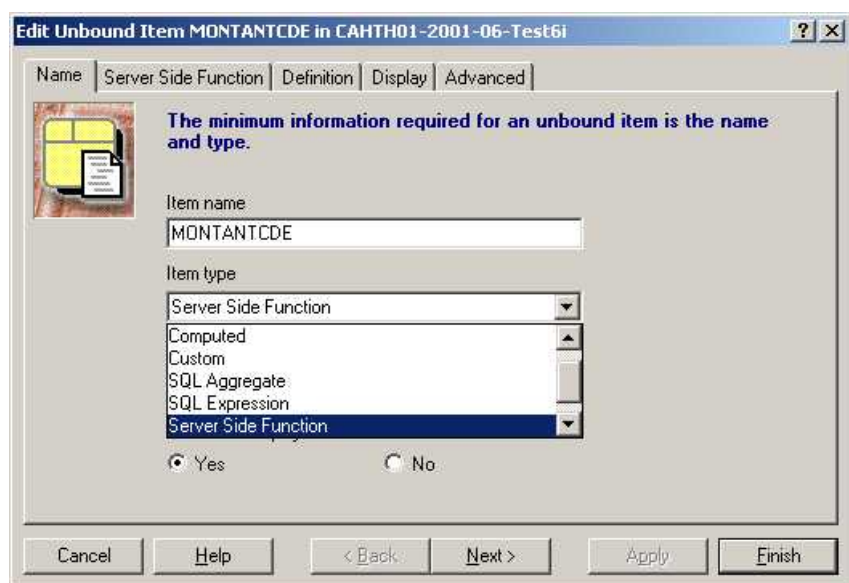


Figure 5 - Type de calcul de champ

3.2.1 Calcul par dérivation

Le champ calculé, **MONTANTLIGNE**, est alimenté par la simple multiplication du prix du produit et de la quantité commandée ; cette règle de calcul se spécifie par l'intermédiaire d'une expression SQL.

SQL	
Derivation Expression	(L_PROD_PRIX_VENTE * QUANTITE)
PL/SQL Block	

Figure 6 - Expression SQL

Le générateur *Web PL/SQL* affecte cette expression SQL au champ calculé **MONTANTLIGNE** lorsqu'il crée la commande SQL de sélection des lignes de commandes.

```
ZONE_SQL := 'SELECT L_PROD_LIBELLE,
                NUMERODEP,
                PROD_NUMERO,
                L_PROD_ID,
                L_PROD_PRIX_VENTE,
                QUANTITE,
                MONTANTLIGNE,
                CDE_NUMERO
            FROM  ( SELECT L_PROD.LIBELLE L_PROD_LIBELLE,
                LIGCDE.NUMERODEP NUMERODEP,
                LIGCDE.PROD_NUMERO PROD_NUMERO,
                L_PROD.ID L_PROD_ID,
                L_PROD.PRIX_VENTE L_PROD_PRIX_VENTE,
                LIGCDE.QUANTITE QUANTITE,
                ( L_PROD.PRIX_VENTE * LIGCDE.QUANTITE) MONTANTLIGNE,
                LIGCDE.CDE_NUMERO CDE_NUMERO
            FROM LIGNESCOMMANDES LIGCDE,
                PRODUITS L_PROD
            WHERE LIGCDE.PROD_NUMERO = L_PROD.NUMERO
            ) ';
```

Figure 7 - Code SQL avec expression

3.2.2 Calcul par une fonction PL/SQL

Le champ calculé, **MONTANTCDE**, doit être alimenté à partir de données qui ne se trouvent pas dans les tables du composant ; dès lors, une simple expression SQL comme précédemment ne peut suffire.

Une première alternative consiste à utiliser une fonction PL/SQL pour alimenter le champ calculé. Cette fonction PL/SQL doit recevoir en paramètre la clé primaire de la commande ; de son côté, la fonction doit rendre comme résultat, la somme du montant de chacune des lignes de la commande.

SQL	
Derivation Expression	PACKCDE.CALCULMONTANTCDE(NUMERO)
PL/SQL Block	

Figure 8 - Appel de fonction PL/SQL

Le générateur *Web PL/SQL* affecte cette fonction PL/SQL au champ calculé **MONTANTCDE** lorsqu'il crée la commande SQL de sélection des commandes.

```

ZONE_SQL := 'SELECT NUMERO,
              CLI_NUMERO,
              DATE_RECEPTION,
              L_CLI_NUMERO,
              L_CLI_NOM,
              L_CLI_PRENOM,
              MONTANTCDE
              FROM ( SELECT CDE.NUMERO NUMERO,
                          CDE.CLI_NUMERO CLI_NUMERO,
                          CDE.DATE_RECEPTION DATE_RECEPTION,
                          L_CLI.NUMERO L_CLI_NUMERO,
                          L_CLI.NOM L_CLI_NOM,
                          L_CLI.PRENOM L_CLI_PRENOM,
                          PACKCDE.CALCULMONTANTCDE(CDE.NUMERO) MONTANTCDE
                    FROM COMMANDES CDE,
                         CLIENTS L_CLI
                    WHERE CDE.CLI_NUMERO = L_CLI.NUMERO
                  ) ' ;

```

Figure 9 - Code SQL avec fonction PL/SQL

3.2.3 Formulaire

Composant d'entête de commandes

Date de réception: 11.01.2000 [CAL](#)

Numéro (client): 2

Nom (client): Dubois

Prénom (client): Anne

Montant total: 543

Mettre à jour Supprimer Rétablir Nouveau

Détail de la commande

Libelle	Id	Prix unitaire	Quantite	Montant
Dérailleur	12.44	199	1	199
Moyeu avant	10.30	134	2	268
Patins de freins	10.32	19	4	76

Nouveau

Nos deux champs calculés sont affichés en lecture dans le formulaire généré par *Web PL/SQL*. En finalité, l'utilisateur dispose d'informations beaucoup plus riches et pertinentes que la simple juxtaposition de données contenues dans les tables des divers composants.

Figure 10 - Formulaire des commandes avec données de montants calculées

4 Table d'agrégat

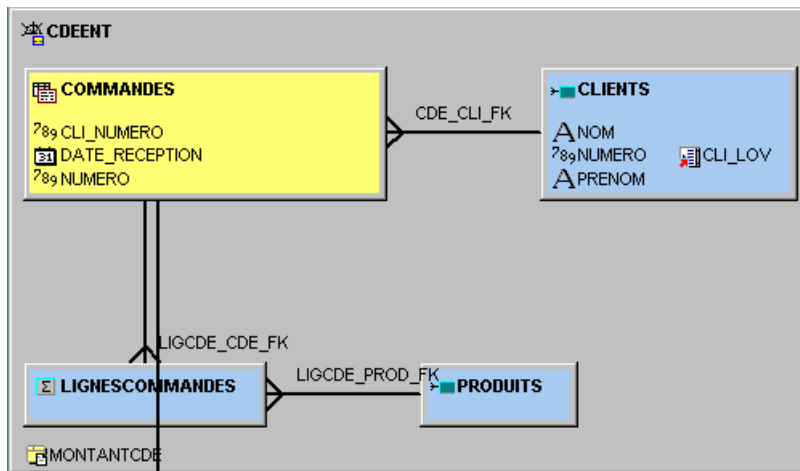


Figure 11 - Utilisation de table d'agrégat

Nous avons vu dans les deux premiers articles, que *Designer* permet de spécifier l'utilisation de tables de base et de tables de références dans un composant ; pour les besoins de fonctions d'agrégation, *Designer* offre la possibilité de définir des tables d'agrégat.

Tout comme pour les liens de maître/détails et de table de référence, les liens d'agrégats sont proposés automatiquement à partir de la structure de données logique, *Server Model*, existante.

La table **LIGNESCOMMANDES** étant intégrée à notre composant en tant que table d'agrégat et la table **PRODUITS** en tant que table de référence de **LIGNESCOMMANDES**, il nous est possible de spécifier une simple expression d'agrégation pour l'alimentation du champ calculé **MONTANTCDE**.

SQL	
Derivation Expression	SUM(LIGCDE.QUANTITE*L_PROD.PRIX_VENTE)
PL/SQL Block	

Figure 12 - Expression SQL avec fonction d'agrégat

```

ZONE_SQL := 'SELECT NUMERO,
                    CLI_NUMERO,
                    DATE_RECEPTION,
                    L_CLI_NUMERO,
                    L_CLI_NOM,
                    L_CLI_PRENOM,
                    PRIXTOTAL
                FROM ( SELECT CDE.NUMERO NUMERO,
                    CDE.CLI_NUMERO CLI_NUMERO,
                    CDE.DATE_RECEPTION DATE_RECEPTION,
                    L_CLI.NUMERO L_CLI_NUMERO,
                    L_CLI.NOM L_CLI_NOM,
                    L_CLI.PRENOM L_CLI_PRENOM,
                    SUM(LIGCDE.QUANTITE*L_PROD.PRIX_VENTE) MONTANTCDE
                FROM COMMANDES CDE,
                    CLIENTS L_CLI,
                    LIGNESCOMMANDES LIGCDE,
                    PRODUITS L_PROD
                WHERE CDE.CLI_NUMERO = L_CLI.NUMERO AND
                    LIGCDE.PROD_NUMERO = L_PROD.NUMERO (+)
                    AND LIGCDE.CDE_NUMERO(+) = CDE.NUMERO
                GROUP BY CDE.NUMERO,
                    CDE.CLI_NUMERO,
                    CDE.DATE_RECEPTION,
                    L_CLI.NUMERO,
                    L_CLI.NOM,
                    L_CLI.PRENOM
                )';
    
```

Le générateur *Web PL/SQL* affecte cette expression SQL au champ calculé **MONTANTCDE** lorsqu'il crée la commande SQL de sélection des commandes ; mais par rapport à la situation précédente, il ajoute les tables **LIGNESCOMMANDES** et **PRODUITS** dans la jointure et la clause d'agrégation **GROUP BY**.

Figure 13 - Code SQL avec fonction d'agrégation

5 Conclusions

Les modules de traitement nous permettent de fournir à l'utilisateur des informations sous forme de données calculées à partir du contenu des tables du modèle logique, *Server Model*.

Nous disposons de plusieurs mécanismes de calcul: des expressions SQL, l'appel de fonctions PL/SQL, l'intégration dans nos composants de tables d'agrégation et d'autres mécanismes que nous n'avons pas présentés dans cet article.

Nous avons aussi vu que l'alimentation de champs calculés peut être spécifiée de différentes manières ; par exemple, le montant total d'une commande peut être calculé par l'appel d'une fonction PL/SQL mais aussi par une utilisation de table d'agrégat.

Entre plusieurs possibilités de calcul, nous préconisons toujours d'utiliser les mécanismes qui apportent un maximum d'automatisme et qui, en contre-partie, minimiseront les travaux de maintenance et les risques de régression lors d'évolution des applications. Pour notre exemple de calcul du montant total de commande, il est évident que nous implanterions l'utilisation de la table d'agrégat; l'utilisation d'une fonction PL/SQL est recommandée seulement dans les cas où un traitement particulier doit être mis en place.

Comme nous l'indiquions en introduction, ces données calculées peuvent aussi être alimentées au niveau du serveur par les APIs de tables ; dans ce cas, le modèle logique de données contiendra des colonnes qui serviront à stocker les données calculées. Ces colonnes introduisent une redondance dans la base de données puisque leur contenu peut toujours être déterminé à partir des données de base.

Dans nos prochains articles, nous présenterons l'alimentation de données calculées par les APIs de tables et compareront cette stratégie à celle que nous venons de présenter.

Toutefois, avant de terminer cet article, il nous semble utile d'établir un bilan des avantages et inconvénients essentiels de la stratégie d'alimentation des données calculées par les modules applicatifs.

Alimentation de données calculées par les applications	
Avantages	Inconvénients /Risques
<ul style="list-style-type: none">• Les informations sont toujours actualisées car elles sont calculées à la volée.• Pas de redondance de données dans la base de données.	<ul style="list-style-type: none">• Les informations doivent toujours être recalculées ce qui peut être pénalisant pour un système à forte charge transactionnelle.• Chaque module applicatif doit mettre en œuvre les mécanismes de calcul, donc, un redondance de travail et un risque de fournir des informations divergentes entre les modules.

6 Liens utiles

L'auteur : Pierre-André Sunier
pierre-andre.sunier@hegne.ch

Site de l'atelier de génie logiciel de la HEG-NE

<http://doc.esnig.ch/>

Rubrique : *Génie logiciel*

Sous-site de formation à Designer

Rubrique : *Génie logiciel / Designer 6i/9i - Les bases*

Sous-site des anciens articles

Rubrique : *Génie logiciel / Publication – Liste des publications / SOUG*

Les cahiers « 03 – Approfondissement des associations » et « 03b – Particularité de modélisation des modules », de notre sous-site de formation *Designer 6i/9i - Les bases*, traitent avec plus de détails pratiques les notions présentées dans cet article.