université PARIS-SACLAY

Bases de données
PL/pgSQL – curseurs

Silviu Maniu

2022/2023

Polytech App3

Contenu

PL/pgSQL

PL/pgSQL : Programming Language for SQL = SQL + langage procédural (impératif)

Rappel :

- **declaratif** (SQL) : dire **quoi**
- **procedural** : dire **comment**

PL/pgSQL : l'union entre SQL et un langage de programmation

PL/SQL – Curseurs

Le probleme :

```
select a,b into x,y from t where cle = 123
```

- récupère l'unique ligne du résultat d'une requête et place dans les variables `x` et `y`
- et si plusieurs lignes ?

PL/pgSQL – Curseurs

Curseur

- zone mémoire
- nommée
- associée à une requête
- peut contenir **0**, **1** ou plusieurs lignes
- taille déterminée à l'exécution
- sert a contenir et traiter l'ensemble des lignes résultat de cette requête

PL/pgSQL – Fonctionnement curseur

Étapes :

1. **déclaration** du curseur
2. **remplissage** en une seule fois par exécution de la requête
3. **récupération** des lignes une par une (parcours séquentiel, pointeur logique)
4. **libération** de la zone

Curseur PL/pgSQL – Exemple (première ligne)

Table `article(refart,nom,prixht)` – récupérer première ligne par ordre alphabétique

- `select into` impossible car plusieurs lignes possibles

```
declare
  c1 cursor for
    select refart from article order by nom;
  art article.refart%type;
begin
  open c1;
  fetch c1 into art; -- affectation de variable (fetch into)
  raise notice '%',art;
  close c1;
end;
```

Curseur PL/pgSQL – Exemple (première et deuxième ligne)

Table `article(refart,nom,prixht)` – récupérer première ligne par ordre alphabétique et la deuxième si elle existe

- variable curseur : `found` – `true` si dernier fetch a trouvé une ligne)

```
declare
  c2 cursor for select refart from article order by nom;
  art article.refart%type;
begin
  open c2; fetch c2 into art; raise notice '%',art;
  fetch c2 into art;
  if found then
    raise notice '%',art;
  end if;
  close c2;
end;
```

Curseur PL/pgSQL – Parcours

Parcours : analogue à un parcours de fichier séquentiel

- pointeur logique

Après `open` :

- pointeur positionné **avant** la première ligne
- `c%found` est `true`

`fetch c into ...` :

- avance le pointeur et lit la ligne pointée
- si pas de ligne, `c%found` devient `false`

Curseur PL/pgSQL – Exemple (tout) 1/2

Table `article(refart,nom,prixht)` – afficher toutes les références

```
declare
  c3 cursor for select refart from article order by nom;
  art article.refart%type;
begin
  open c3; fetch c3 into art;
  while found loop
    raise notice '%',art;
    fetch c3 into art;
  end loop;
  close c3;
end;
```

Curseur PL/pgSQL – Exemple (tout) 2/2

Table `article(refart,nom,prixht)` – afficher toutes les références et le prix

```
declare
  c4 cursor for select refart, prixht from article;
  art record; -- contient les champs résultat
begin
  open c4; fetch c4 into art;
  while found loop
    raise notice '% %',art.refart,art.prixht;
    fetch c4 into art;
  end loop;
  close c4;
end;
```

Curseur PL/pgSQL – Utilisation boucles (1/2)

Peut utiliser aussi `loop ... exit when not found` (elimine redondance de `fetch`) :

```
declare
  c5 cursor for select refart, prixht from article;
  art record;
begin
  open c5;
  loop
    fetch c5 into art;
    exit when not found;
    raise notice '% %',art.refart,art.prixht;
  end loop;
  close c5;
end;
```

Curseur PL/pgSQL – Utilisation boucles (2/2)

Forme la plus elegante `for x in c ...`

- elimine `open`, `close`
- elimine la necessité de `fetch`

```
declare
  c5 cursor for select refart, prixht from article;
  art record;
begin
  for art in c6 loop
    raise notice '% %',art.refart,art.prixht;
  end loop;
end;
```

Remerciements

Le contenu de ce cours est grandement inspiré des cours d'Emmanuel Waller

- `https://www.lri.fr/~waller/`