

# Base de données

## Cours 9 - Procédures, fonctions et import/export

Steve Lévesque, Tous droits réservés © où applicables

# Table des matières

- 1 Procédures stockées
  - Création et exécution
  - Paramètres et variables
- 2 Fonctions
  - Fonctions scalaires
  - Fonctions tableaux
- 3 Importation et exportation
  - Exportation de données
  - Importation de données
  - Sauvegarde et restauration

# Procédures stockées

Les procédures stockées permettent d'encapsuler des séquences d'opérations SQL complexes dans des routines réutilisables, exécutables côté serveur pour une meilleure performance et maintenance.

# Procédures stockées - Création et exécution

Il est possible de créer des procédures stockées sans paramètres, en utilisant le langage SQL ou PL/pgSQL.

```
1 CREATE OR REPLACE PROCEDURE update_course_enrollments()
2 LANGUAGE SQL
3 AS $$
4     -- Update enrollment dates for all records
5     UPDATE student_course SET enrollment_date = CURRENT_DATE
6     WHERE enrollment_date IS NULL;
7
8     -- Insert audit record
9     INSERT INTO audit_log (action_type, action_date)
10    VALUES ('Enrollment date update', NOW());
11 $$;
12 -- Execute the procedure
13 CALL update_course_enrollments();
```

Listing: <https://www.postgresql.org/docs/current/sql-createprocedure.html>

# Procédures stockées - Paramètres et variables

```
1  -- Create a procedure with parameters for student enrollment
2  CREATE OR REPLACE PROCEDURE enroll_student(IN p_student_id INT)
3  LANGUAGE plpgsql
4  AS $$
5  BEGIN
6      -- Check if student exists
7      IF NOT EXISTS (SELECT 1 FROM student WHERE student_id = p_student_id)
8      THEN
9          RAISE EXCEPTION 'Student with ID % does not exist', p_student_id;
10     END IF;
11
12     -- Enroll student in course with id 1
13     INSERT INTO student_course (student_id, course_id, enrollment_date)
14     VALUES (p_student_id, 1, CURRENT_DATE);
15     COMMIT;
16 EXCEPTION
17     WHEN OTHERS THEN
18         ROLLBACK;
19         RAISE;
20 END;
21 $$;
```

Listing: <https://www.postgresql.org/docs/current/sql-createprocedure.html>

# Fonctions

Les fonctions permettent de créer des routines réutilisables qui retournent une valeur, utilisables dans des expressions SQL pour modulariser le code et centraliser la logique.

# Fonctions - Fonctions scalaires

Une fonction scalaire **retourne une valeur unique** (un scalaire) et **n'est exécutée qu'une seule fois par ligne traitée**. Le type de retour d'une fonction scalaire peut être un nombre, une chaîne de caractères, une date, etc.

Elles peuvent être utilisées dans des expressions SQL, des clauses SELECT, WHERE, etc.

# Fonctions - Fonctions scalaires

```
1  -- Scalar function to calculate total students in a department
2  CREATE OR REPLACE FUNCTION count_department_students(
3      dept_id INT
4  ) RETURNS INT
5  LANGUAGE SQL
6  STABLE
7  AS $$
8      SELECT COUNT(DISTINCT student_course.student_id)
9      FROM student_course
10     JOIN course ON student_course.course_id = course.course_id
11     WHERE course.department_id = count_department_students.dept_id;
12  $$;
13
14  -- Usage in query
15  SELECT department_name, count_department_students(department_id)
16  AS student_count
17  FROM department;
```

Listing: <https://www.postgresql.org/docs/current/sql-createfunction.html>

# Fonctions - Fonctions tableaux

Une fonction tableau **retourne un ensemble de lignes (une table)** plutôt qu'une valeur unique. Elle peut être **utilisée comme une table dans une requête SQL**.

# Fonctions - Fonctions tableaux

```
1  -- Function returning table of department courses with enrollment count
2  CREATE OR REPLACE FUNCTION get_department_courses(
3      dept_id INT
4  ) RETURNS TABLE ( course_code VARCHAR(20), course_name VARCHAR(100),
5                      instructor VARCHAR(100), student_count BIGINT)
6  LANGUAGE plpgsql
7  AS $$
8  BEGIN
9      RETURN QUERY
10     SELECT c.course_code, c.course_name, c.instructor,
11            COUNT(sc.student_id) AS student_count
12     FROM course c
13     LEFT JOIN student_course sc ON c.course_id = sc.course_id
14     WHERE c.department_id = get_department_courses.dept_id
15     GROUP BY c.course_id, c.course_code, c.course_name, c.instructor;
16 END;
17 $$;
18 -- Usage of table function
19 SELECT * FROM get_department_courses(1);
```

Listing: <https://www.postgresql.org/docs/current/sql-createfunction.html>

# Importation et exportation

L'importation et l'exportation de données sont essentielles pour migrer des données, faire des sauvegardes, ou échanger des données avec d'autres systèmes.

# Importation et exportation - Exportation de données

Il est possible d'exporter des données en formats texte ou binaire, à l'aide de la commande COPY ... TO.

```
1  -- Export all students to CSV
2  COPY student TO '/tmp/students_export.csv' WITH CSV HEADER;
3
4  -- Export course data in PostgreSQL binary format
5  COPY course TO '/tmp/courses_export.bin' WITH BINARY;
```

Listing: <https://www.postgresql.org/docs/current/sql-copy.html>

Il est aussi possible d'exporter des données en format personnalisé.

```
1  -- Export student enrollments in custom format
2  COPY student_course TO '/tmp/enrollments_export.txt'
3  WITH DELIMITER '|' NULL 'NULL';
```

Listing: <https://www.postgresql.org/docs/current/sql-copy.html>

# Importation et exportation - Exportation de données

Nous pouvons également exporter des données spécifiques en utilisant des requêtes SQL.

```
1  -- Export specific department courses with enrollment count
2  COPY (
3      SELECT d.department_name, c.course_code, c.course_name,
4             COUNT(sc.student_id) AS enrollment_count
5      FROM course c
6      JOIN department d ON c.department_id = d.department_id
7      LEFT JOIN student_course sc ON c.course_id = sc.course_id
8      WHERE d.department_id = 1
9      GROUP BY d.department_name, c.course_code, c.course_name
10 ) TO '/tmp/department1_courses.csv' WITH CSV HEADER;
```

Listing: <https://www.postgresql.org/docs/current/sql-copy.html>

# Importation et exportation - Importation de données

Il est possible d'importer des données en formats texte ou binaire, à l'aide de la commande COPY ... FROM.

```
1  -- Import new students from CSV file
2  COPY student FROM '/tmp/new_students.csv' WITH CSV HEADER;
3
4  -- Import with specific column mapping
5  COPY student (first_name, last_name)
6  FROM '/tmp/student_names.csv' WITH CSV HEADER;
7
8  -- Import course data from binary format
9  COPY course FROM '/tmp/courses_import.bin' WITH BINARY;
```

Listing: <https://www.postgresql.org/docs/current/sql-copy.html>

# Sauvegarde et restauration - Sauvegarde

La sauvegarde crée une copie des données à un moment dans le temps, pour pouvoir les récupérer plus tard. La sauvegarde de bases de données est cruciale pour la récupération après sinistre, la migration de données, et la maintenance régulière.

# Sauvegarde et restauration - Sauvegarde

Il est possible de faire des sauvegardes complètes, des sauvegardes de schéma uniquement, ou des sauvegardes de données uniquement, à l'aide de l'outil `pg_dump`.

```
1  -- Full database backup (terminal command)
2  pg_dump -h localhost -U username -d school -F c -f school_complete.dump
3
4  -- Schema-only backup
5  pg_dump -h localhost -U username -d school -s -f schema_only.dump
6
7  -- Data-only backup (no schema)
8  pg_dump -h localhost -U username -d school -a -f data_only.dump
```

Listing: <https://www.postgresql.org/docs/current/app-pgdump.html>

# Sauvegarde et restauration - Restauration

La restauration est le processus qui utilise la sauvegarde pour remettre les données dans un état antérieur connu et sain.

# Importation et exportation - Restauration

Il est possible de restaurer une base de données complète, ou de restaurer à partir d'une sauvegarde SQL simple, à l'aide des outils `pg_restore` et `psql`.

```
1  -- Complete restore (terminal command)
2  pg_restore -h localhost -U username -d school school_complete.dump
3
4  -- Restore with database creation
5  -- createdb school
6  pg_restore -h localhost -U username -d school school_complete.dump
7
8  -- Plain SQL backup
9  pg_dump -h localhost -U username -d school -F p -f school.sql
10
11 -- Plain SQL restore
12 psql -h localhost -U username -d school -f school.sql
```

Listing: <https://www.postgresql.org/docs/current/app-pgrestore.html>

# Bibliographie

- <https://www.postgresql.org/docs/current/sql-createprocedure.html>
- <https://www.postgresql.org/docs/current/sql-createfunction.html>
- <https://www.postgresql.org/docs/current/sql-copy.html>
- <https://www.postgresql.org/docs/current/app-pgdump.html>
- <https://www.postgresql.org/docs/current/app-pgrestore.html>
- <https://www.postgresql.org/docs/current/app-psql.html>