Facultad de Ingeniería

Introducción a la programación en base de datos

Tema VIII

Semestre 2024-1

UNAM
UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Objetivo

El alumno explicará el uso de las extensiones de la base de datos y aplicará los elementos necesarios para la creación de programas en lenguaje procedural para procesar y gestionar la información contenida en la base de datos.

Características

Los lenguajes de programación en BD, no forman parte del estándar SQL.

Extensiones

SQL fue diseñado para un propósito específico: Realizar consultas dentro de una base de datos.

¿Y si quiero ir más allá de sólo consultas?

Extensiones

Los distintos manejadores proporcionan extensiones de SQL para añadir funcionalidad. En el caso de postgres es PL/pgSQL

Otros lenguajes

Existe la posibilidad de poder emplear algún lenguaje de programación dentro del manejador, además de las extensiones SQL.

Otros lenguajes

Varían entre cada manejador. En el caso de postgres, podemos emplear python, C, perl, etc.

Como todo lenguaje de programación, consta de una gramática.

Variables

Dentro de cualquier programa se tienen dos módulos:

- **Declaraciones**
- **Cuerpo**

Variables

Dentro del módulo de declaraciones, podemos establecer las variables que vamos a usar

```
DECLARE var1 INT DEFAULT 10;
DECLARE var2 char(4) = 'Hola';
```

También se cuenta con estructuras de control similares a las de los lenguajes comunes, como if’s, for’s, case’s, etc.

Funciones

Ente matemático que, dado un argumento, regresa algo.

La idea es la misma respecto a otros lenguajes de programación.

Funciones

```
CREATE [OR REPLACE] FUNCTION nombre ([argumentos])
RETURNS tipo_dato AS $$
—seccion_Declaraciones
BEGIN
— cuerpo_Funcion
[RETURN valor]
END;
$$
LANGUAGE PGSQL;
```

Triggers

Elemento de programación en BD que ejecuta una función en un momento determinado cuando un evento ocurre.

3 tipos:

- **BEFORE**
- **AFTER**
- **INSTEAD OF**

Triggers

```
CREATE [ CONSTRAINT ] TRIGGER name { BEFORE | AFTER | INSTEAD OF } { event [ OR ... ] }
    ON table_name
    [ FROM referenced_table_name ]
    [ NOT DEFERRABLE | [ DEFERRABLE ] [ INITIALLY IMMEDIATE | INITIALLY DEFERRED ] ]
    [ REFERENCING { { OLD | NEW } TABLE [ AS ] transition_relation_name } [ ... ] ]
    [ FOR [ EACH ] { ROW | STATEMENT } ]
    [ WHEN ( condition ) ]
    EXECUTE { FUNCTION | PROCEDURE } function_name ( arguments )
```

Triggers

Puede ser aplicado sobre las siguientes sentencias DML:

- **INSERT**
- **UPDATE**
- **DELETE**

Triggers

2 modos de ejecución:

- FOR EACH STATEMENT
- FOR EACH ROW

Triggers

Imaginen una tabla “t” con millones de registros y lanzan lo siguiente:

```
UPDATE t
SET edad = edad + 1;
```

Triggers

Un trigger crea variables especiales, veremos algunas.

Procedures

- A partir de la versión 11

¿Sintaxis? ¿Si no la recuerdo?

- **Una diferencia notoria es que los procedures tienen la característica del control de transacciones.**