SQL\_PLSQL

# SQL

## Types

CREATE OR REPLACE TYPE *nom\_type* AS OBJECT (*var1 type1*, *var2 type2, var3 type3*); 🡪 Crée un type d’objet

CREATE OR REPLACE TYPE *nom\_type* AS TABLE OF *nom\_type2*; 🡪 Créé une liste d’objets

# PLSQL

## Variables

var typeVar := typeVar (att1, att2, att3);

## Fonctions et procédures

Chaque fonction et procédure PL/SQL peut être constitué de 3 sections :

* [**DECLARE** … section facultative de déclaration et initialisation]
* **BEGIN** … section obligatoire contenant les instructions exécutables
* [**EXCEPTION** … section facultative gérant les erreurs]  
  **END;**

|  |
| --- |
| CREATE OR REPLACE **FUNCTION** nom\_fonction(param1 IN type\_param1, param2 IN type\_param2…)  **RETURN** type\_retourne **IS**  DECLARE  Var1 typeVar1;  Var2 typeVar2;  BEGIN  --Code ;  **RETURN** var\_a\_retourner;  END; |

|  |
| --- |
| CREATE OR REPLACE **PROCEDURE** nom\_fonction(param1 IN type\_param1, param2 IN type\_param2…)  **AS**  DECLARE  Var1 typeVar1;  Var2 typeVar2;  BEGIN  --Code ;  END; |

## Conditions

|  |
| --- |
| **If** condition **then** *code*  **else** *code*  **end if**; |

## Autre

**exec** (fonc\_proc) 🡪 Exécuter une fonction ou une procédure

**commit** 🡪 mettre a jour la BD après avoir utilisé un update

**insert into** *table* **values** (*att1*, *att2*, *attx*) 🡪 ajouter un tuple dans table

**select** *XXX* **into** *var* **from** *bdd* 🡪 récupérer les données d’une requête de bdd dans var

# TD1

Depuis sqldevelopper

## Partie 1

|  |
| --- |
| CREATE OR REPLACE TYPE Coordonnee AS OBJECT  (Ville VARCHAR(50), CP VARCHAR(5), Tel VARCHAR2, Fax VARCHAR2); |



|  |
| --- |
| CREATE OR REPLACE TABLE EMP  (  NUMEMP INTEGER PRIMARY KEY,  NOM\_COMPLET VARCHAR(25),  Salaire INTEGER,  Coordonnees Coordonnee  ); |



|  |
| --- |
| INSERT INTO EMP values (14, 'DUPUIS Yvonne', NULL, Coordonnee ('Lyon', '69000', 0472546585, null));  INSERT INTO EMP values (15, 'DUPOND Jean', null, Coordonnee ('Paris', '75000', null, null));  INSERT INTO EMP values (16, 'CRINIERE Belle', null, Coordonnee ('Grenoble', '38001', null, null));  INSERT INTO EMP values (17, 'AUBERT Louis', null, Coordonnee ('Lyon', '69100', 0478556585, null));  INSERT INTO EMP values (18, 'MAURI John', null, Coordonnee ('Anglet', '64200', null, null)); |

|  |
| --- |
| UPDATE EMP e SET e.salaire = 30000 WHERE e.numemp = 2;  UPDATE EMP e SET e.salaire = 20000 WHERE e.numemp = 3; |

|  |
| --- |
| UPDATE EMP E SET salaire = &salaire WHERE NUMEMP = &numEmp ; |



|  |
| --- |
| SELECT DISTINCT e.coordonnee.Ville FROM EMP e |



|  |
| --- |
| SELECT COUNT (DISTINCT(e.Coordonnee.Ville)) |



|  |
| --- |
| SELECT \* FROM EMP e WHERE e.Coordonnee.Ville = 'Anglet'; |



|  |
| --- |
| SELECT \* FROM EMP E WHERE E.Coordonnee.Telephone IS NOT NULL ; |



|  |
| --- |
| DECLARE  coordonneeXav Coordonnee := Cordonnee ('Anglet', 64600, 0638454545,0638444545);  BEGIN  UPDATE EMP SET Coordonnee = CordonneeXav WHERE NOM\_COMPLET = 'XAVIER Richard;  END |

## Partie 2

|  |
| --- |
| CREATE OR REPLACE PROCEDURE **INSERT\_EMP** (p1 IN NUMBER, p2 IN VARCHAR2, p3 IN NUMBER, p4 in VARCHAR2, p5 in VARCHAR2, p6 in VARCHAR2, p7 in VARCHAR2)  AS  DECLARE  coord := new Coordoonnee(p4, p5, p6, p7) ;  BEGIN  INERT INTO EMP VALUES (p1, p2, p3, coord) ;  COMMIT;  END; |

|  |
| --- |
| CREATE OR REPLACE PROCEDURE **INSERT\_EMP** (nom IN VARCHAR2, salaire IN NUMBER(7))  AS  BEGIN  UPDATE EMP SET salaire = sal WHERE nomComplet = nom ;  END; |

|  |
| --- |
| CREATE OR REPLACE FUNCTION **COUNT\_EMP** ()  RETURN NUMBER IS  DECLARE  compteur NUMBER  BEGIN  SELECT COUNT(\*) INTO compteur FROM EMP  RETURN compteur;  END; |

|  |
| --- |
| CREATE OR REPLACE FUNCTION **COUNT\_EMP** (ville IN VARCHAR2)  RETURN NUMBER IS  DECLARE  compteur NUMBER  BEGIN  SELECT COUNT(\*) INTO compteur FROM EMP WHERE Coordonnee.ville = &ville ;  RETURN compteur;  END; |

# TD2

## Partie 1

1. Création du type Point

|  |
| --- |
| CREATE OR REPLACE TYPE Point AS OBJECT (x NUMBER(5), y NUMBER(5)) ; |

1. Création du type Ligne

|  |
| --- |
| CREATE OR REPLACE TYPE Ligne AS OBJECT (p1 Point, p2 Point) ; |

1. Création du type Polygone

|  |
| --- |
| CREATE OR REPLACE TYPE Polygone AS TABLE OF Point ; |

1. Création de la table REGION

|  |
| --- |
| CREATE OR REPLACE TABLE REGION (nom VARCHAR2, coord Polygone) ; |

1. Ajout de l’Aquitaine dans la table REGION

|  |
| --- |
| DECLARE  p Polygone ;  BEGIN  p := Polygone(Point(1,1)) ;  INSERT INTO REGION VALUES (‘Aquitaine’, p) ;  END; |

1. Création du type Rectangle

|  |
| --- |
| CREATE OR REPLACE TYPE Rectangle AS OBJECT (p1 Point, p2 Point) ; |

1. Création de la table VILLE

|  |
| --- |
| CREATE OR REPLACE TABLE VILLE (nom VARCHAR2, rectangle Rectangle); |

1. Ajout de tuples dans la table VILLE

|  |
| --- |
| INSERT INTO VILLE VALUES (‘Anglet, Rectangle(Point(8950, 4840), Point(9180, 5050)));  INSERT INTO VILLE VALUES (‘Lyon, Rectangle (Point(8820, 7020), Point(9080, 7230)));  INSERT INTO VILLE VALUES (‘Paris, Rectangle (Point(6310, 2940), Point(6560, 3110))); |

1. Affichage des villes et de leur surfaces dans VILLE

|  |
| --- |
| SELECT nom, (Rectangle.p2.x - Rectangle.p1.x) \* (Rectangle.p2.y - Rectangle.p1.y)  FROM VILLE |

1. Affichage des villes ayant un des côtés > 10 dans VILLE

|  |
| --- |
| SELECT nom FROM VILLE  WHERE (Rectangle.p2.x - Rectangle.p1.x) > 10 OR  (Rectangle.p2.y - Rectangle.p1.y) > 10 |

## Partie 2

1. Création de la table

|  |
| --- |
| CREATE OR REPLACE TYPE Metadonnees AS OBJECT(  nomAuteur VARCHAR2,  geolocalisation NUMBER,  dateExpiration DATE) ;  CREATE OR REPLACE TYPE Lien AS OBJECT(  url VARCHAR2,  relation VARCHAR2)  CREATE OR REPLACE TYPE Liens AS TABLE OF Lien ;  CREATE OR REPLACE TABLE RO (  url VARCHAR2,  texte VARCHAR2,  objet MIME,  date DATE,  metadonnees Metadonnees,  liens Liens) ; |

1. Insertion d’un tuple

|  |
| --- |
| INSERT INTO RO VALUES(‘’Ma-photo’’,  ‘’Photo prise lors d’une soirée avec les amis’’,  12/05/09,  Metadonnees(‘’X. Tintin’’, 43.834 – 0.352, 03/04/08),  Liens(Lien(‘’Ma-photo2’’, ‘’thème’’), Lien(‘’Ma-photo12’’, réponse))) ; |