TP BD 1

Marc Gagné

1er décembre 2015

Ci-dessous se trouve une collection de commandes utilisées lors du TP BDR 1. Elles ne sont pas toutes de la plus grande qualité, mais devraient toutes être syntaxiquement correctes.

1 Historique des commandes

```
ALTER TABLE Country
  ADD CONSTRAINT COUNTRY_CAPITALKEY
    FOREIGN KEY (Code, Capital, Province)
    REFERENCES City (Country, Name, Province);
ALTER TABLE City
  ADD CONSTRAINT CITY_PROVINCEKEY
    FOREIGN KEY (Country, Province)
    REFERENCES Province (Country, Name);
ALTER TABLE Province
  ADD CONSTRAINT PROVINCE_COUNTRYKEY
    FOREIGN KEY (Country)
    REFERENCES Country (Code);
ALTER TABLE Borders
  ADD CONSTRAINT BORDERS_LENGTHPOSITIVE
    CHECK (LENGTH > 0) ENABLE;

ALTER TABLE Country
  MODIFY (Name NOT NULL);
ALTER TABLE Country
  ADD CONSTRAINT COUNTRY_UNIQUENAME
    UNIQUE (NAME);
ALTER TABLE Economy
  ADD CONSTRAINT ECONOMY_PERCENTAGE
    CHECK (
      AGRICULTURE >= 0 AND
      AGRICULTURE <= 100 AND
      SERVICE >= 0 AND
      SERVICE <= 100 AND
      INDUSTRY >= 0 AND
      INDUSTRY <= 100
    ) ENABLE;

INSERT INTO Country
  (Name, Code, Area, Population)
  VALUES ('Atlan', 'AT', 400, 180000);
INSERT INTO Country
  (Name, Code, Area, Population)
  VALUES ('Tis', 'TIS', 100, 20000);
```

```
INSERT INTO Province
   (Name, Country, Population, Area)
   VALUES ('Atlan', 'AT', 400, 180000);
INSERT INTO Province
   (Name, Country, Population, Area)
   VALUES ('Tis', 'TIS', 100, 20000);

UPDATE Country
   SET Capital=NULL, Province=NULL
   WHERE Code='ATL';
UPDATE Province
   SET Capital=NULL
   WHERE Name='Atlantis' AND Country='ATL';
UPDATE City
   SET Country='AT', Province='Atlan'
   WHERE Name='Atlantis City' AND Country='ATL' AND Province='Atlantis';

DELETE FROM Province WHERE Name='Atlantis' AND Country='ATL';
DELETE FROM Country WHERE Code='ATL';

ALTER TABLE PROVINCE
   ADD CONSTRAINT PROVINCE_CAPITALKEY
      FOREIGN KEY (Name, Country, CapProv)
      REFERENCES City (Province, Country, Name);

/* 2.3.1 */
SELECT * FROM Country ORDER BY Name ASC;

/* 2.3.2 */
SELECT Organization, COUNT(*) AS Members FROM Is_Member
   WHERE Type='member'
   GROUP BY Organization
   HAVING COUNT(*)=(
      SELECT MAX(COUNT(*)) FROM Is_Member
         WHERE Type='member' GROUP BY Organization
   );

/* 2.3.3 */
SELECT c1.Code, c1.Name, SUM(c2.Population/c1.Population) CityPopulation
   FROM Country c1, City c2
   WHERE c2.COUNTRY = c1.Code
   GROUP BY c1.Code, c1.Name
   ORDER BY CityPopulation DESC;

/* 2.3.4 */
SELECT Continent, SUM(c.Population*e.PERCENTAGE/100) Population
   FROM Encompasses e, Country c
   WHERE e.Country = c.Code
   GROUP BY Continent
   ORDER BY Population ASC;

/* 2.3.5 */
CREATE OR REPLACE VIEW v_235 AS(
```

```
  SELECT Country1 Country, SUM(Length) Length
  FROM Borders
  GROUP BY Country1
);
INSERT INTO v_235
  SELECT Country2 Country, SUM(length) Length
  FROM Borders
  GROUP BY Country2;
SELECT Country, Length
  FROM v_235
  ORDER BY Length DESC;

/* 2.3.6 */
SELECT Name, COUNT(*)
  FROM City
  GROUP BY Name
  HAVING COUNT(DISTINCT Country) > 1;

/* 2.3.7 */
SELECT Name, COUNT(*)
  FROM City
  GROUP BY Name
  HAVING COUNT(DISTINCT Country) > 1
  ORDER BY COUNT(*) DESC, Name;

/* 2.3.8 */
SELECT Code, Name
  FROM Country c, Politics p
  WHERE c.Code = p.Country AND p.Independence IS NULL;

/* 2.3.9 */
SELECT Code, Name
  FROM Country
  WHERE Code NOT IN (
    SELECT DISTINCT Country
      FROM Is_Member
      WHERE Country IS NOT NULL
  );

/* 2.3.10 */
SELECT c.Code, c.Name, m.Organization
  FROM Country c, Is_Member m
  WHERE c.Code = m.Country(+)
  ORDER BY c.Code;

/* 2.3.17 */
CREATE OR REPLACE PROCEDURE p_2317(
  p$code IN Country.Code%TYPE,
  p$name OUT country.name%TYPE,
  p$nbFront OUT NUMBER
)
IS
BEGIN
```

```
  SELECT Name
    INTO p$name
    FROM Country
    WHERE Code = p$code;

  SELECT COUNT(*)
    INTO p$nbFront
    FROM Borders
    WHERE Country1 = p$code
    OR Country2 = p$code;

END;

-----------

CREATE OR REPLACE PROCEDURE TEST_p_2317(
  p$code IN Country.Code%TYPE
)
IS
  v$name Country.Name%TYPE;
  v$nbFront NUMBER;
BEGIN
  p_2317(p$code, v$name, v$nbFront);

  dbms_output.put_line(
    'Le pays de code ' ||
    p$code ||
    ' a pour nom ' ||
    v$name ||
    ' et pour nombre de frontieres ' ||
    v$nbFront
  );
END;
/
show errors;

-----------

execute TEST_p_2317('F');

/* 2.4 */
SELECT TABLE_NAME FROM user_tables;

/* 2.5.1 */
CREATE OR REPLACE TRIGGER TRG_Politics_CheckIndependence
  BEFORE INSERT OR UPDATE ON Politics
  FOR EACH ROW
BEGIN
  IF (:new.Independence > SYSDATE())
  THEN
    RAISE_APPLICATION_ERROR(-20000, 'Invalid Independence Day.');
  END IF;
END;
```

```
/* 2.5.2 */
CREATE OR REPLACE TRIGGER TRG_Country_Code
  BEFORE INSERT OR UPDATE ON Country
  FOR EACH ROW
BEGIN
  IF (NOT REGEXP_LIKE(:NEW.Code, '^[A-Za-z]+$'))
  THEN
    RAISE_APPLICATION_ERROR(-20001, 'Invalid Code.');
  END IF;
  :NEW.Code := UPPER(:NEW.Code);
END;

/* ********************** TP 2 ************************** */

-- 1.1
CREATE OR REPLACE VIEW Countries_100M AS
  SELECT * FROM Country
  WHERE Population > 100000000;

-- 1.2
INSERT INTO Countries_100M
  (Name, Code, Area, Population)
  VALUES ('Selmana', 'SMA', 780000000, 1000000000);

SELECT * FROM Countries_100M;
SELECT * FROM Country;

-- 1.3
INSERT INTO Countries_100M
  (Name, Code, Area, Population)
  VALUES ('Thibautie', 'TBT', 2, 20);

SELECT * FROM Countries_100M;
SELECT * FROM Country;

-- 1.4
CREATE OR REPLACE VIEW Countries_Europe AS
  SELECT Name, Code, Capital, Province, Area, Population, Percentage
    FROM Country JOIN Encompasses
    ON Country.Code = Encompasses.Country
    WHERE Continent = 'Europe';

INSERT INTO Countries_Europe
  (Name, Code, Area, Population, Percentage)
  VALUES ('Fictivia', 'FIC', 1000, 1000, 100);

-- 1.5
CREATE OR REPLACE VIEW CountryOrganization AS
  SELECT m.Country AS Country,
         o.Abbreviation AS Organization,
         o.Name AS OrganizationName,
         o.City AS OrganizationCity,
```

```
        o.Country AS OrganizationCountry,
        o.Province AS OrganizationProvince,
        m.Type AS MembershipType
    FROM Is_Member m, Organization o
    WHERE m.Organization = o.abbreviation;

CREATE OR REPLACE TRIGGER TRG_CountryOrganization_Insert
  INSTEAD OF INSERT ON CountryOrganization
  REFERENCING NEW AS n
  FOR EACH ROW
BEGIN
  INSERT INTO Organization
    (Abbreviation, Name, City, Country, Province)
    VALUES (
      :n.Organization,
      :n.OrganizationName,
      :n.OrganizationCity,
      :n.OrganizationCountry,
      :n.OrganizationProvince
    );
  INSERT INTO Is_Member
    (Country, Organization, Type)
    VALUES (:n.Country, :n.Organization, :n.MembershipType);
END;

INSERT INTO CountryOrganization
  (
    Country,
    Organization,
    OrganizationName,
    OrganizationCity,
    OrganizationCountry,
    OrganizationProvince,
    MembershipType
  )
  VALUES ('F', 'B3309', 'Binome B3309', 'Montreal', 'CDN', 'Quebec', 'founder');
```

2 Script de génération des GRANT

```
SET HEADING OFF
SET ECHO OFF
SET FEEDBACK OFF
SPOOL '\\servif-home\binomes\IF-B3309\insa\3if\tp\tp-bd_1\grant.sql'

SELECT
 'GRANT SELECT ON ' || Table_Name || ' TO snemmaoui,tcolard;\n' ||
 'GRANT UPDATE ON ' || Table_Name || ' TO snemmaoui,tcolard;' AS Commands
FROM User_Tables;

SPOOL OUT

@\\servif-home\binomes\IF-B3309\insa\3if\tp\tp-bd_1\grant.sql
```