Semester 2 Project
Instructions for Students

Project Setup: The Data

This project will use a case study called TRAVELER ASSISTANCE. A set of database tables is used to provide information and manage the world travelers' requests. Information is stored about all the countries in the world and their languages, currencies, demographics, and the region in which they are located. This database will assist travelers to obtain specific information about the countries to which they wish to travel.

The database objects for this project will need to be created by running the PLSQL_Schema.sql script located in this project interaction - and they are as follows:

COUNTRIES, REGIONS, CURRENCIES, SPOKEN_LANGUAGES and LANGUAGES tables.

General Programming Guidelines:

- Code your SQL statements to query the data regardless of whether it is stored in upper or lower case (use the LOWER and/or UPPER functions).
- Include an exception handler to handle NO_DATA_FOUND and other relevant exceptions in all your procedures.

The Assignment and Deliverables:

Create the following programs:

- Part 1: traveler_assistance_package
- Part 2: traveler_admin_package

Part 1: Provide Basic Information to Travelers

Create a package called *traveler_assistance_package* that will contain the following six procedures. Make all procedures public. Comment your procedures to explain their purpose and functionality.

Two procedures in this package (*countries_in_same_region* and *country_languages*) return their fetched data back to the calling environment as an OUT parameter which is an associative array (ie an INDEX BY table of records). The last two procedures (*print_region_array* and *print_language_array*) will accept and display the returned arrays.

1. Create a procedure called *country_demographics* to display specific information about a country.
   - Pass COUNTRY_NAME as an IN parameter. Display COUNTRY_NAME, LOCATION, CAPITOL, POPULATION, AIRPORTS, CLIMATE. Use a user-defined record structure for the INTO clause of your select statement. Raise an exception if the country does not exist.

   Hints:
   - In order to populate the record in the select statement without specifying the record components, the record structure must be identical to the column list on the select statement.

2. Create a procedure called *find_region_and_currency* to fetch and return the currency and region in which a country is located.
   - Pass COUNTRY_NAME as an IN parameter and use a user-defined record as an OUT parameter that returns the country name, its region and currency.

   Hints:
   - Declare a user-defined record TYPE in the package spec with appropriate components. Use this record type to declare record variables in your procedure.

3. Create a procedure *countries_in_same_region* to fetch and return all the countries in the same region.
   - Pass REGION_NAME as an IN parameter and a PLSQL associative array of records (an INDEX BY table) as an OUT parameter. Return REGION_NAME, COUNTRY_NAME, and CURRENCY_NAME through the OUT parameter for all the countries in the requested region.

   Hints:
   - The OUT parameter should be an associative array of records.
   - Declare an associative array of records TYPE in the package spec and use this type declaration to declare the OUT parameter.
   - The *print_region_array* procedure will display the contents of the array returned by the OUT parameter.

4. Create a procedure *print_region_array* to display the content of an array of records that is passed to it.
   - Pass an associative array of records that was declared in procedure *countries_in_same_region*, as an IN Parameter. The procedure should display its content.

5. Create a procedure *country_languages* to fetch and return all the spoken language(s) and the official language(s) for a country.
   - Pass COUNTRY_NAME as an IN parameter. The OUT parameter is an associative array that will return COUNTRY_NAME, LANGUAGE_NAME and OFFICIAL.
     Note: A country may have multiple spoken languages. A country may also have more than one official language. Check the OFFICIAL field in SPOKEN_LANGUAGES table to obtain the official languages for a country.

   Hints:
   - Create a PLSQL associative array of record TYPE in the package spec. You can use this data type to declare the OUT parameter in the procedure.
   - The *print_language_array* procedure will display the contents of the array returned by the OUT parameter.

6. Create a procedure *print_language_array* to display the content of an array of records that is passed to it.
   - Pass an associative array of records that was declared in procedure *countries_languages*, as an IN Parameter. The procedure should display its content.

Part 2: Traveler System Administration

Create a package called *traveler_admin_package*, which can be used to maintain the system.

1. Create a procedure *display_disabled_triggers* that displays a list of all disabled triggers in your schema.

2. Create a function *all_dependent_objects* that returns all the dependent objects for a particular object.
   - Pass OBJECT_NAME as an IN parameter and return an array that contains the NAME , TYPE, REFERENCED_NAME AND REFERENCED_TYPE values.

   Hints:
   - Query the data dictionary and RETURN an associative array of records from the body of the function.
   - If a function returns an empty array, an ORA-06502 exception will be raised. Include code to test whether the associative array contains at least one record; if it does not, populate the first field of the first record with a suitable message.

3. Create a procedure *print_dependent_objects* that displays the array of dependent objects returned by the *all_dependent_objects* function.