Technical Report RT-123

Relational database development: conceptual and physical models with creation scripts

André Plante and Sylvain Martin

February 2003

For reference:

Plante, A. and S. Martin (2003). Relational database development: conceptual and physical models with creation scripts. Technical Report MSC Québec – Hydrology RT-123, Environment Canada, Sainte-Foy, prepared for the Plan Formulation and Evaluation Group (PFEG) of the of the International Lake Ontario – St. Lawrence River Study Board (International Joint Commission). 24 pages + Appendix.

RESEARCH TEAM

Environment Canada, **Meteorological Service of Canada – Hydrology Section.**

Coordination, design and report writing	André Plante, Sylvain Martin
Programming	Sylvain Martin
Revision	André Bouchard

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES

1 INTRODUCTION

The Lake Ontario and the St. Lawrence River Study of the International Joint Commission (IJC) will be generating a substantial amount of data/information/knowledge. The Common Data Needs Technical Working Group (CDNTWG) of the International Joint Commission's Lake Ontario – St. Lawrence River Study was charged with the development and implementation of an Information Management Strategy (IMS). With the assistance of a contractor, Pangaea Information Technologies, the IMS team has conducted a comprehensive Needs Assessment (NA) and hosted two workshops to aid in the formulation of the IMS (Pangaea, 2001). The implementation of the IMS was to be done by the Information Management Task Work Group (IMTWG). The CDNTWG was therefore transformed in the IMTWG.

The Pangaea Study concluded that the best method for achieving the goals of the study in terms of Data Storage, Maintenance, Access, and Distribution was to use regionally distributed database management systems. Under this option, interoperability standards need to be respected to ensure integration and connectivity to other systems, and accommodate other technologies such as geospatial web services.

Consequently, this document presents conceptual and physical data models aiming to implement and store the list of Common Geographical Information System (GIS) Base Layers as described by the CDNTWG in its Short-term GIS Guidelines and that many Technical Work Groups identified as necessary to perform analyses. The model developed within this project aims to respect the proposition issued from the partnership between CDNTWG and Great Lakes geospatial data community members to include the development of a framework for essential geospatial data in the IMS goals. Specific CDNTWG project objectives were fulfilled while these goals were achieved.

The work presented here was done within the scope of a project that has for main objectives to produce the necessary data structure to support the 7 base layers of the Information management strategy (IMS) as well as defining the conceptual basis for one of the regional relational databases that will support the IMS. The present work can however be adapted to the other nodes easily. This project involves the Meteorological Service of Canada (MSC-Quebec, Hydrology section), the Great Lakes Commission, Land Information of Ontario and the International Joint Commission (IJC).

The proposed model follows the pattern of a typical relational database with the implementation of entities (layers) and relational constraints between entities.

2 Information Management Strategy

2.1 Vision

The Information management strategy (IMS) favours data use, reuse and sharing. To do so, two considerations must be addressed by the IMS: First the data must be managed in a way as to facilitate public outreach and second, data access by study participants must be as effective as possible to permit quick execution of their work.

2.2 Design rules

To meet the IMS vision outlined above, the following design rules will be followed to produce the conceptual and physical data models;

- This work must be done in respect of a global architecture which must be flexible enough to accommodate the heterogeneous nature of the data produced by the entire study and be scalable to permit growth as the study progresses.
- The database must be able to hold the data at every stage of its life cycle (values, data, and knowledge).
- The data must be available to all users and study participants not withstanding the type of tools being used. Study participants should not have to modify their business processes in order to participate in this study. Users should be able to use the tools they know to perform the needed analysis.
- Each piece of data contained in the database must be accompanied by its metadata.
- Maximum use of the internet is necessary to address the large variety of users; general public, researchers and study participants.
- Automated distribution, which includes such things as spatial web services, must be supported again to favour the maximum use of the internet.
- The work done for designing this data model must be documented to permit easy maintenance and technology transfer.

3 Conceptual Data Model

Beyond many belief and techniques that follow fashion, conceptual data modeling consists in a rigorous representation of what is stable and hidden behind the apparent variety which constructs our reality. This is captured in a model, which is a representation of reality, using tables, identification attributes and relationships.

Data to be captures in this Conceptual Data Model

The CDNTWG (2002) produced a guideline document to help integration of data throughout the entire study area. Seven data layers where describe in this report are available to all groups and allow all data to be referenced to the same base layers. Table 1 summarizes the seven base layers with a short description of each. An additional layer, which was not described in the CDNTWG (2002), was added to hold cadastral information. This was deemed necessary by the Technical Implementation comitee.

Table 1: Base layer name and description.

Base layer name	**Base layer description**
Shoreline	Line of intersection of land and water for a defined water elevation
Political Units	Municipal and county boundaries
Transportation Features	Roads within 5 km buffer of shoreline
Watersheds	All watershed boundaries within Lac Ontario and the St-Lawrence River basins
Tributaries	All tributaries within the above basins
DEM	Digital elevation model of floodplain and river bed
Conservation Management Units	Conservation areas within the Lac Ontario and the St-Lawrence River basins
Cadastral information	Description of the land registry in the study area

Tables

A total of height data topics must be captured in the present data modeling exercise. Two of the layers, transportation features and hydrologic basins, should be taken directly from the data producer using

their web services. The data producer for both of these layers is NRCAN and efforts will be done in that respect. The benefits here are three-fold: first the most up-to-date information will be used, second this approach prevents data duplication and third the IMTWG does not have to manage this information.

In order to store data for each of the six remaining data topics depicted by the CDNTWG, twelve basic tables are proposed in the database. Thus table SHORELINE, POLITICAL_UNIT, TRIBUTARY, LIDAR, GRID, BATHYMETRY, CONTOUR, CMU (Conservation Management Unit), CADAS_POLY and CADAS_INFO are proposed to store data from the base layers.

The two last tables are common to all data layers to be produced as part of this study and are used to hold descriptive information. The first one is a table that stores data type definition for all layers of data to be produced by the study and is named TYPE_DEFINITION. This table is necessary to qualify the information in a per line approach. This approach was chosen because it permits the definition of an infinite number of data types. The second one is the table METADATA_CATALOG whose function is to associate metadata information with each piece of data in the datasets. Figure 1 shows all the tables as well as the relational constraints between these tables.

Descriptive and identification attributes

Each table contains fields who's sole purpose is to contain attributes that describe and identify the data values held in the database. These fields are Type_id, which describes the data type and Dats_id which associates metadata information on what they represent.

In order to describe a data type, three attributes are used in the table TYPE_DEFINITION. First, the attribute Type_id will supply each data type with a unique identification number. Next, attributes Type_name and Type_description will respectively store the name given to the data type and a short definition.

The metadata information associated which each piece of data is contained in the table named METADATA_CATALOG. This table contains a unique identifier and the location and name of the XML file which contains the metadata produced by the responsible authority for the data.

Next, the POLITICAL_UNIT table will contain the political units information layer. For each unit, the attribute Unit_id identifies each political unit, Unit_name will store the political unit name, Boundary stores the polygons describing the unit (series of connected points and projection/datum information) and finally the attributes Type_EN and Type_FR store the type of political unit in English and French respectively (for example in English: county, state, municipal).

The table TRIBUTARY will store information about the tributary layer. In this table, each tributary is uniquely identified by the attribute Tributary_id. The watershed in which it resides is identified by Watershed_id and its name is stored in the attribute Name. Finally, the tributary itself is stored as a geometric object in the Tributary attribute (series of points and projection/datum information).

The Digital Elevation Model (DEM) layer is stored in the database using four tables. The first one contains the LIDAR raw data and each point that is part of the model is individually stored into the LIDAR table. The point itself (X, Y and Z values and projection/datum information) is stored in the attribute Point. The Point_id attribute identifies each point of the model individually. The second table contains a regular 5 meter grid where the attributes point_id and point have the same role than in the previous table. An additional field, Flag, is an evaluation of the significance of the point (Technical report 120, Fortin et al., 2002). The third table contains shallow water bathymetric data. It is also composed of the attribute point_id and point. Finally, the last table contains 4 meter-interval contour lines. This table is supplied with the attributes Line_id to identify each line, Line that stores the line itself (series of points and projection/datum) and elevation to store the height of the isoline.

The layer that models the Conservation Management Unit (CMU) is stored in the CMU table of the database. The attribute CMU_id will uniquely identity each CMU. This table is also built with the attributes CMU_name to respectively store the CMU name and subunit name. The attribute Desig is the one in which is stored the designation of the CMU and the attributes Cat_IUCN and Creation are used store the CMU IUCN category (http://www.iucn.org/themes/forests/6/notitle.html) and the CMU creation date respectively. Finally, the polygon describing each CMU subunit (series of connected points and projection/datum) is stored in the attribute CMU.

Finally, Cadastral information are going to stored into two tables. The first one will store spatial information and is named CADAS_POLY as polygons are used to model land registry geometry. The other table, CADAS_INFO will store non-spatial information using a “per line” approach to take into account different land registries from different region.

Relational Constraints

In the conceptual model, a relational constraint is shown by an arrow which starts from the identifying attribute in the child table and ends in the mother table at the attribute of the same name. The table containing all data types definition (TYPE_DEFINITION) is the mother table and the data layer table is the child table. The attribute that uniquely identifies data types in the mother table is a primary key in the mother table. The same attribute appears in the data layer table where it plays the role of a foreign key. In order to link this table to the mother table, the layer table must also have this identification to refer to the table where the data type is explained. The same type of relation exists between each layer table and the table METADATA_CATALOG. Therefore, each of the layer tables have relations with two other tables, TYPE_DEFINITION and METADATA_CATALOG.

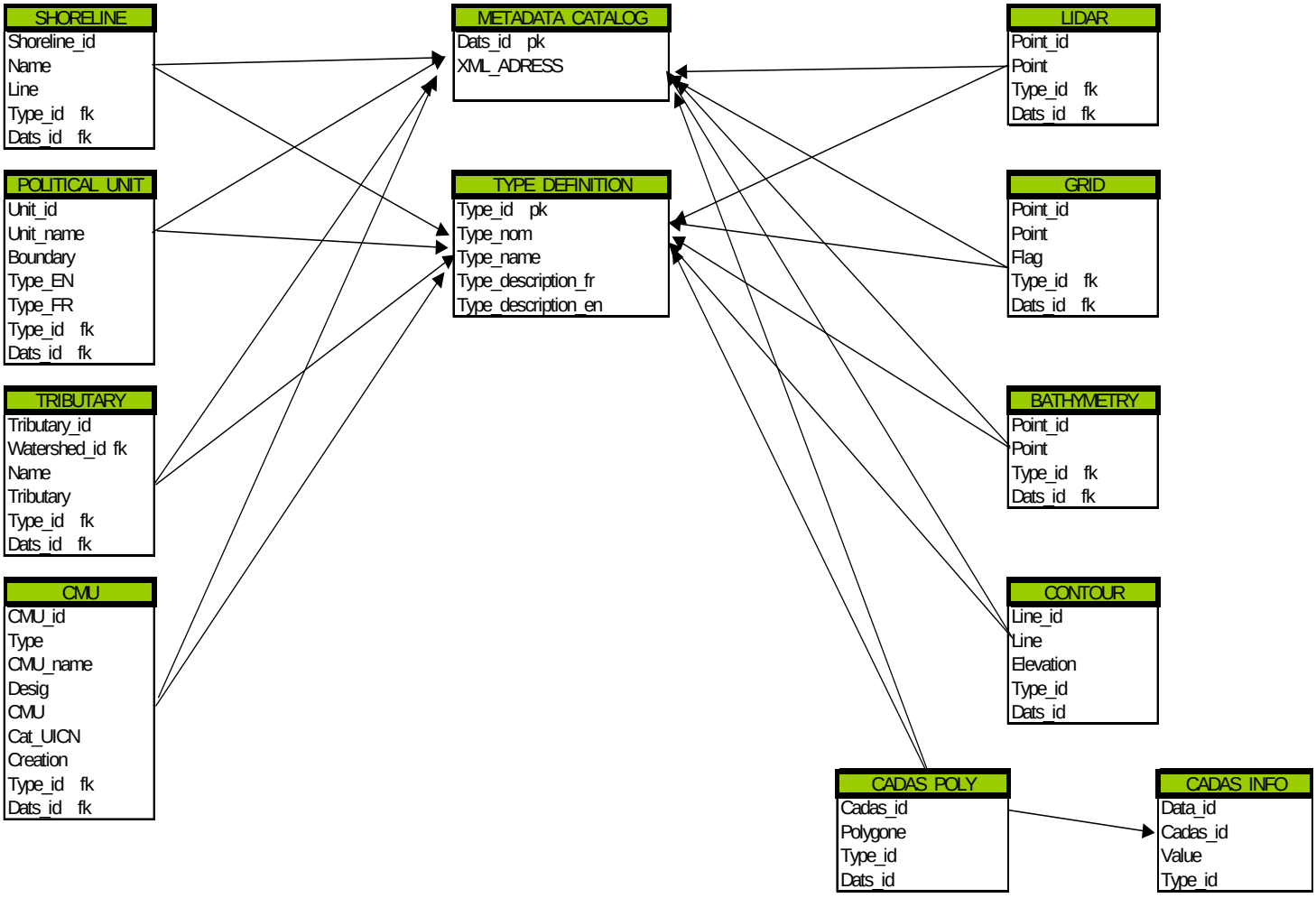

Figure 1: Conceptual data model describing the relational database.

4 Physical Data Model

In order for the relational database to contain, in an organized fashion, the layers described in the short-term GIS guidelines produced by the Common Data Needs Technical Work Group (CDNTWG), tables are used. The physical data model is derived from the conceptual data model. It, in turn, will be used to generate the necessary creation scripts. For each table, attributes are presented with a description of their data type and length. Other characteristics are also shown like primary or foreign keys and the null possibility (absent value). A description of each attribute is done and associated

constraints are detailed. The tables are presented in their order of creation within the database. This is necessary to respect relational constraints among tables.

Data types used here follow the Oracle 9i SQL data types implementation. The four following types were used:

Number: Numerical type (real or integer according to the size and precision)

Varchar2: Variable length character chain type.

Mdsys.sdo_geometry: Oracle implementation for spatial objects (point, line, polygon, circle, polygon with hole, multipolygon etc.)

Date: Oracle type for storing a date.

4.1 Table TYPE_DEFINITION

This table contains a description of all the different types of data that are held in each of the layers present in the database. Each of the tables of the model contains a foreign key on the attribute Type_id. For this reason, this table is created first.

Atribute	**Type**	**Length**	**Null**	**Indexed**	**Key**
Type_id	Number	3	No	Yes	Primary
Type_nom	Varchar2	25	No	No	
Type_name	Varchar2	25	No	No	
Type_description_fr	Varchar2	250	No	No	
Type_description_en	Varchar2	250	No	No	

Attribute description:

Type_id: Unique identification number for a data type.

Type_name: The name of the data type in English.

Type_nom: The name of the data type in French.

Type_description_fr: Short description of the data type in French.

Type_description_en: Short description of the data type in English.

Constraints:

- Each type_id must be unique.

4.2. Table METADATA_CATALOG

Each table contains a foreign key on the attribute Dats_id from the table METADATA_CATALOG. Before one inserts a data set table, an entry must appear in the METADATA_CATALOG table that refers to metadata information. This insures that each piece of data in the database is linked with it's metadata.

Atribute	**Type**	**Length**	**Null**	**Indexed**	**Key**
Dats_id	Number	3	No	Yes	Primary
XML_ADRESS	Varchar2	100	No	No	

Attribute description:

Dats_id: Unique identification number for a data type.

XML_ADRESS: Adress of the XML file in which is described this datatype.

Constraints:

- Each Dats_id must be unique.

4.3 SHORELINE Layer

Table SHORELINE contains information pertaining to the St. Lawrence River shoreline. These are constituted of lines and are inserted in this table as spatial objects.

Atribute	**Type**	**Length**	**Null**	**Indexed**	**Key**
Shoreline_id	Number	10	No	Yes	Primary
Name	Varchar2	50	No	No	
Line	Mdsys.sdo_geometry		No	Yes	
Type_id	Number	10	No	No	Foreign
Dats_id	Number	10	No	No	Foreign

Attribute description:

Shoreline_id: Unique identification number for a shoreline

Name: Water-course name.

Line: Spatial object to represent a shoreline.

Type_id: Data type identification number.

Dats_id: Metadata identifier.

Constraints:

- Each Shoreline_id number must be unique.
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

4.4 POLITICAL_UNIT Layer

This table can contain any kind of political unit. These are represented as polygons representing territorial sub-divisions and are stored as spatial objects. The name of the type of political unit is also taken into account for both French and English.

Atribute	Type	Length	Null	Indexed	Key
Unit_id	Number	10	No	Yes	Primary
Unit_name	Varchar2	50	No	No	
Boundary	Mdsys.sdo_geometry		No	Yes	
Type_EN	Varchar2	20	Yes	No	
Type_FR	Varchar2	20	Yes	No	
Type_id	Number	10	No	No	Foreign
Dats_id	Number	10	No	No	Foreign

Attribute description:

Unit_id: Unique identification number for each political unit.

Unit_name: Name of the political unit.

Boundary: Spatial object (polygon) to represent the boundaries of a political unit.

Type_EN, Type_FR: Type of political unit.

Type_id: Data type identification number.

Dats_id: Metadata identifier.

Constraints:

- Each Unit_id number must be unique.

- Type_EN must be 'MUNICIPAL' or 'COUNTY' or 'STATE' or 'PROVINCIAL' (can be updated to accept more unit types).
- Type FR must be 'MUNICIPAL' or 'CONTE' or 'ETAT' or 'PROVINCE' or 'FEDERALE' (can be updated to accept more unit types).
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

4.5 TRIBUTARY Layer

This table contains the name of the tributaries that flow into the St. Lawrence River. They are stored as spatial objects into the attribute Tributary. The Watershed_id is the watershed identification number or character chain that will be obtained from the supplier and that can be used as the primary key within this database.

Atribute	**Type**	**Length**	**Null**	**Indexed**	**Key**
Tributary_id	Number	10	No	Yes	Primary
Watershed_id	Number	10	No	Yes	
Name	Varchar2	50	No	No	
Tributary	Mdsys.sdo_geometry		No	Yes	
Type_id	Number	10	No	No	Foreign
Dats_id	Number	10	No	No	Foreign

Attribute description:

Tributary_id: Unique identification number for each Tributary

Watershed_id: Identification number of the watershed in which resides the tributary, obtained from NRCAN - Canadian Digital Drainage Areas

Name: Name of the tributary

Tributary: Spatial object (line) to represent a tributary

Type_id: Data type identification number

Dats_id: Metadata identifier.

Constraints:

- Each Tributary_id number must be unique.

- The watershed_id must exist.
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

4.6 DEM (Digital Elevation Model) Layer

The DEM is based on two types of data, the light detection and ranging (LIDAR) data and shallow water bathymetry data. Both of these data sets were acquired by the EC-Québec hydrology section. A total of four datasets therefore compose the DEM data and will require four separate tables: LIDAR raw data, LIDAR on a regular 5 meter grid, shallow water bathymetric data and 4 meter interval contour lines.

The first table contains the raw LIDAR data covering the St-Lawrence River floodplain between Beauharnois and Trois-Rivières. Each table row contains a point resulting from the LIDAR campaign.

Atribute	**Type**	**Length**	**Null**	**Indexed**	**Key**
Point_id	Number	10	No	Yes	Primary
Point	Mdsys.sdo_geometry		No	Yes	
Type_id	Number	10	No	No	Foreign
Dats_id	Number	10	No	No	Foreign
Flag	Number	2	No	Yes	

Attribute description:

Point_id: Unique identification number for each point of the LIDAR.

Point : Spatial object (point) to represent each point of the LIDAR.

Flag : Parameter indicating the height variation around a point.

Type_id : Data type identification number.

Dats_id: Metadata identifier.

Constraints:

- Each Point_id number must be unique.

- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

The second table contains the 5 meter square grid data with average height values. This table was generated from the LIDAR values. The EC-Québec hydrology section produced a flag value to permit extraction of LIDAR values based on significance of the topographic features. This flag value and the method used to produce it is described in the report that accompanied that deliverable (Technical report 120, Fortin *et al*., 2002).

Atribute	**Type**	**Length**	**Null**	**Indexed**	**Key**
Point_id	Number	10	No	Yes	Primary
Point	Mdsys.sdo_geometry		No	Yes	
Flag	Number	10, 4	No	No	
Type_id	Number	10	No	No	Foreign
Dats_id	Number	10	No	No	Foreign

Attribute description:

Point_id: Unique identification number for each point on the regular 5 meter grid.

Point: Spatial object (point) to represent each point of the regular 5 meter grid.

Flag: Maximum height variation among neighbouring grid points.

Type_id: Data type identification number.

Dats_id: Metadata identifier.

Constraints:

- Each Point_id number must be unique.
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

The third table will contain the shallow water bathymetry data gathered at some selected areas on the St-Lawrence. Technical Report no. 118 produced by Fortin (2002) describes this data set.

Atribute	Type	Length	Null	Indexed	Key
Point_id	Number	10	No	Yes	Primary
Point	Mdsys.sdo_geometry		No	Yes	
Type_id	Number	10	No	No	Foreign
Dats_id	Number	10	No	No	Foreign

Attribute description:

Point_id: Unique identification number for each point of the shallow water bathymetry.

Point: Spatial object (point) to represent each point of the shallow water bathymetry.

Type_id : Data type identification number.

Dats_id: Metadata identifier.

Constraints:

- Each Point_id number must be unique.
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

The fourth and last table contains contour data for the area from Cornwall to Trois-Rivière. The procedures used to generate this DEM are covered in the report which accompanied this delivrable(Technical report 120, Fortin et al., 2002). The generation of these contours uses the LIDAR data, the bathymetry data obtained in the shallow water work done and the bathymetry in the numerical model developed by the hydrology section. This data was projected on a the 5 meter grid to be used in contour generation.

Atribute	Type	Length	Null	Indexed	Key
Line_id	Number	10	No	Yes	Primary
Line	Mdsys.sdo_geometry		No	Yes	
Elevation	Number	23, 16	No	No	
Type_id	Number	10	No	No	Foreign
Dats_id	Number	10	No	No	Foreign

Attribute description:

Line_id: Unique identification number for each line.

Line: Spatial object (line) to represent each contour line.

Elevation: Height of the isoline.

Type_id : Data type identification number.

Dats_id: Metadata identifier.

Constraints:

- Each Line_id number must be unique.
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

4.7 CMU (Conservation Management Unit) Layer

This table models Conservation Management Units (CMU) and contains essential information to store these CMUs. As such, CMUs are stored as spatial objects (polygons). Oracle Spatial implements functions designed to calculate area and perimeter, and consequently, no attributes are used to store these parameters.

A CMU may be composed of many subunits (for example, a park having many islands) and as such may be stored as a multipolygon spatial object type.

Atribute	**Type**	**Length**	**Null**	**Indexed**	**Key**
CMU_id	Number	10	No	Yes	Primary
Cmu_name	Varchar2	75	No	No	
Desig	Varchar2	75	No	No	
CMU	Mdsys.sdo_geometry		No	Yes	
Cat_UICN	Varchar2	20	No	No	
Creation	Date		No	No	
Type_id	Number	10	No	No	Foreign
Dats_id	Number	10	No	No	Foreign

Attribute description:

CMU_id : Unique identification number for each CMU.

CMU_name: Name of the CMU.

Desig: CMU designation.

CMU: Spatial object (polygon) to represent a CMU.

Cat_UICN: Represents one of the six categories or subcategories created by the world conservation union to describe Protected areas. Visit this site: http://www.iucn.org/themes/forests/6/notitle.html for more information.

Creation: CMU's creation date.

Type_id: Data type identification number.

Dats_id: Metadata identifier.

Constraints:

- Each CMU_id number must be unique.
- CAT_IUCN must be a valid category (Ia, Ib, II, III, IV, V, VI).
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

4.8 Cadastral information Layer

Cadastral information is to be stored into two tables, the first one, CADAS_POLY, stores a cadastral unique identifier for the associated polygon, the polygon itself and a metadata reference. The other table, CADAS_INFO, stores non-spatial attributes that are related to a polygon. As stored cadastral information may vary from a region to another, it has been chosen to store this information in a manner for all data to be stored. Each data type related to cadastral information has to be identified as a data type in the TYPE_DEFINITION table. Each information that has to be stored in the CADAS_INFO table has to be inserted with the type_id that corresponds to the data type inserted into the TYPE_DEFINITION table; it also has to be inserted with the cadastral identifier so that this information is associated with the right polygon.

Table CADAS_POLY

Atribute	**Type**	**Length**	**Null**	**Indexed**	**Key**
Cadas_id	Number	10	No	Yes	Primary
Polygone	Mdsys.sdo_geometry		No	Yes	
Type_id	Number	10	No	No	Foreign
Dats_id	Number	10	No	No	Foreign

Attribute description:

Cadas_id : Unique identification number for each CMU.

Polygone: Spatial object (polygon) to represent a CMU.

Type_id: Data type identification number.

Dats_id: Metadata identifier.

Constraints:

- Each Cadas_id number must be unique.
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

- The foreign key Dats_id refers to the primary key Dats_id in the METADATA_CATALOG table.

Table CADAS_POLY

Atribute	**Type**	**Length**	**Null**	**Indexed**	**Key**
Data_id	Number	10	No	Yes	Primary
Cadas_id	Number		No	Yes	
Value	Number	23, 16	No	No	
Type_id	Number	10	No	No	Foreign

Attribute description:

Data_id: Unique identification number for each stored data.

Cadas_id: Cadastral identifier.

Value: Stored value for the corresponding data type.

Type_id: Data type identifier number.

Constraints:

- Each Data_id number must be unique.
- The foreign key Cadas_id refers to the primary key Cadas_id in the CADAS_POLY table.
- The foreign key Type_id refers to the primary key Type_id in the TYPE_DEFINITION table.

5 Database Physical Model Creation Scripts

Creation scripts are necessary to transfer the structure produced through the conceptual and physical data model work. These are written in SQL-DDL using a normal text editor. Table 1 depicts creation script names and their individual roles. . Script no 1 creates the logical and physical spaces where all the tables and indexes are going to be stored. The next script creates the user USR_CDN who is going to create all the layer tables.

Tableau 2: Creation scripts name and role in the database creation.

No	Creation script name	Role
1-	SC_TA_TABLESPACE.SQL	Creates tablespace
2-	SC_USERS.SQL	Creates user USR_CDN
3-	SC_TA_TYPE_DEF.SQL	Creates table TYPE_DEFINITION
4-	SC_TA_METADATA_CATALOG.SQL	Creates table METADATA_CATALOG
5-	SC_TA_SHORELINE.SQL	Creates table SHORELINE
6-	SC_TA_POLITICAL_UNIT.SQL	Creates table POLITICAL_UNIT
7-	SC_TA_TRIBUTARY.SQL	Creates table TRIBUTARY
8-	SC_TA_DEM.SQL	Creates table DEM
9-	SC_TA_CMU.SQL	Creates table CMU
10-	SC_TA_CADASTRAL_INFO.SQL	Creates tables for Cadastral infomation
11-	SC_PROJECTION.SQL	Inserts short-term guideline projection

General documentation

RELATIONS

Each of the tables used to describe the layers contains a unique data type. These data types are described in the TYPE_DEFINITION table of the database. This creates a “relationship” between each table representing the data layers and the TYPE_DEFINITION table. In a relational model context, this is implemented with a primary key in the mother table, represented here by the attribute type_id in the TYPE_DEFINITION table; and a foreign key in the child tables, represented also by the attribute type_id in each of the different layer tables.

UNIQUE CONSTRAINTS

Some of the tables have a unique constraint on their identification number attribute (ex. Shoreline_id has a unique constraint). This is used to avoid duplication. A sequence generator is created for each of these attributes to help create the values given to

identification numbers as unique values. Additionally, each insertion in these tables must use the NEXTVAL pseudocolumn. The next SQL sentence inserts a row in the table SHORELINE. This table uses the Sequence SHORELINE_SEQ.

INSERT INTO TABLE USR_CDN.SHORELINE VALUES (SHORELINE_SEQ.NEXTVAL, 'St.Lawrence river shore', MDSYS.SDO_GEOMETRY (2002, 82196…….

ATTRIBUTE CONSTRAINTS

Three attributes in all of the tables have constraints, Type_EN and Type_FR in the POLITICAL_UNIT table and CAT_IUCN in the CMU table. These constraints are used to avoid wrong values for these attributes. For example, the constraint CAT_IUCN_CST checks if the value entered in the CAT_IUCN column is an existing protected area category as defined by the IUCN. If the value does not correspond to a valid category, the insertion is blocked. The attributes Type_EN and Type_FR in the POLITICAL_UNIT follow the same rule.

SPATIAL INDEX MAINTENANCE

As the quality of spatial indexes may decrease as spatial data are inserted in a table, it is useful to rebuild these indexes after data insertion. To do this, one can use the command:
ALTER INDEX *index_name* REBUILD.
This command can be submitted to the database with the SQL*PLUS, SQL*plusworksheet or TOAD software environment.

CREATION SCRIPT ORDER

Tables are created by existing users. Therefore users have to be created before the table since tables become part of a user's schema. Next, in order to respect relational constraints, mother tables (table containing information on which other tables depend) have to be created before child tables. So, the tables TYPE_DEFINITION and METADATA_CATALOG have to be created before all the layer tables. The creation script for table TYPE_DEFINITION also inserts the rows necessary in this table so that it has enough information for other tables to be created and populated. A total of 8 rows is being added for this purpose.

The creation order suggested in table 1 respects relational constraints and user creation.

PROJECTION/DATUM

All the scripts for layer table creation insert a line in the MDSYS.SDO_GEOM_METADATA view indicating to Oracle which projection is used to store the spatial attribute. The spatial reference system used here is the Lambert Conformal Conic which has been created and inserted into the database before running the creation scripts. An additional script is supplied (SC_PROJECTION.SQL) along with creation scripts that permits the insertion of a new coordinate system into an Oracle database with parameters corresponding to the Lambert Conformal Conic Projection specification specified in the GIS short term guidelines document produced by the CDNTWG. Its spatial reference identification number (SRID) 3000000 in this example.

Oracle implements nearly a thousand referencing systems and the line inserted into the MDSYS.SDO_GEOM_METADATA view is specific for the projection/datum Lambert Conformal Conic/NAD 83. Other specific values have to be inserted into the view to use another referencing system or one can use a transformation function from Oracle Spatial to use spatial objects in a different projection/datum context than the one into which it has been stored.

CREATION SCRIPT FEATURES

Each creation script drops every object before recreating them. This way, if one has to run a given script more than once, objects don't have to be dropped individually before running the script again. However, if data is present in the table being dropped, it will be deleted.

At the beginning of each table creation script, the user is granted sufficient rights to create the table. These rights are revoked at the end of each script. This way, one can connect with the user name but cannot create objects.

VIEWS AND TABLES

For each layer table, a view is created and gives access to all table attributes with the exception of the Type_id and Dats_id. These views avoid end-users seeing management information and also avoid data updates by end users. Users may have different rights on a view or a table. Some users have the right to select, or insert, or delete or update. For example, the data set manager can select, insert, delete or update rows in a table or a view, but an end-user may only have the right to select information on a view. In this case, end-users will have selection rights on created views only, to avoid any unwanted updates on tables.

6 Conclusion

The work accomplished within this report includes:

- The description of a vision and associated design objectives for the Information Management Strategy (IMS).
- The production of conceptual and physical data models that form the basis for the entire database. A per line conceptual approach was used because of the expandability and scalability this approach permits.
- The production of the SQL scripts needed to create the database as well as the tables and relational constraints needed to organize the different data sets covered in the short term GIS guidelines of the CDNTWG.

As the study progresses, the following changes to the database are expected to occur:

- More data types will be added to the Type_Definition table.
- More rows will be added to the Metadata_Catalog table.
- More tables will be created within the database to cover the data sets produced by the different Task Work Groups (TWG) as they complete their work.
- Relational Constraints will be added to the data model.
- Adjustments to the data model to meet study objectives will be needed.

7 References

Fortin, P., S. Martin et A. Plante (2002). Post-traitement, validation et intégration des données LIDAR dans le modèle numérique de terrain du fleuve Saint-Laurent. Environnement Canada, Service météorologique du Canada, Rapport technique RT-120, Sainte-Foy. 47 p.

Fortin, P. (2002). Acquisition and post-processing of bathymetric data in shallow waters for the section of the St. Lawrence River between Cornwall and Trois-Rivières. Environnement Canada, Service météorologique du Canada, Rapport technique RT-118, Sainte-Foy. 34 p.

Pangeae (2002). Information Management Strategy for the International Joint Commission Lake Ontario-St.Lawrence River Study, May 2002. Pangeae Information Technologies, Ltd. 130 p.

CDNTWG (2002). Short-Term GIS Guidelines. Produced by the Common Data Need Task Work Group of the International Joint Commission Lake Ontario-St. Lawrence River Study, March 21, 2002. 7 p.

APPENDIX A – CREATION SCRIPTS

For each creation script, a text file is supplied and contains SQL commands for object creation. These commands are also documented with comments accompanying them. AS they are small documents, they were copied on a 3.5 inches diskette. The following documents appear on the diskette:

1- SC_TA_TABLESPACE.SQL
2- SC_USERS.SQL
3- SC_TA_TYPE_DEF.SQL
4- SC_TA_METADATA_CATALOG.SQL
5- SC_TA_SHORELINE.SQL
6- SC_TA_POLITICAL_UNIT.SQL
7- SC_TA_TRIBUTARY.SQL
8- SC_TA_DEM.SQL
9- SC_TA_CMU.SQL
10- SC_TA_CADASTRAL_INFO.SQL
11- SC_PROJECTION.SQL

1- SC_TA_TABLESPACE.SQL

```
--*****************************************************
--*              CREATION SCRIPT
--*                 FOR TABLESPACE
--*****************************************************

CONNECT SYS/*****@***** AS SYSDBA;

--*********************************************************
--*        TABLESPACE DESTRUCTION
--*********************************************************

DROP TABLESPACE TS_INDEX3 INCLUDING CONTENTS AND DATAFILES;
DROP TABLESPACE TS_CDN INCLUDING CONTENTS AND DATAFILES CASCADE CONSTRAINTS;
DROP TABLESPACE TS_COMMUN3 INCLUDING CONTENTS AND DATAFILES  CASCADE CONSTRAINTS;
DROP TABLESPACE TS_TEMP3 INCLUDING CONTENTS AND DATAFILES;



--*********************************************************
--*        TABLESPACE CREATION
--*********************************************************
CREATE TABLESPACE TS_INDEX3 DATAFILE 'C:\ORADATA\TS_INDEX3.DAT' SIZE 2 M
          AUTOEXTEND ON MAXSIZE UNLIMITED
          EXTENT MANAGEMENT LOCAL UNIFORM SIZE 128 K;

CREATE TEMPORARY TABLESPACE TS_TEMP3 TEMPFILE 'C:\ORADATA\TS_TEMP3.DAT' SIZE 5 M REUSE
          AUTOEXTEND ON MAXSIZE UNLIMITED;

CREATE TABLESPACE TS_COMMUN3 DATAFILE 'C:\ORADATA\TS_COMMUN3.DAT' SIZE 1 M
          AUTOEXTEND ON MAXSIZE UNLIMITED
          EXTENT MANAGEMENT LOCAL UNIFORM SIZE 100 K;

CREATE TABLESPACE TS_CDN DATAFILE 'C:\ORADATA\TS_CDN3.DAT' SIZE 10 M
          AUTOEXTEND ON MAXSIZE UNLIMITED
          EXTENT MANAGEMENT LOCAL UNIFORM SIZE 128K;


PROMPT *******************************
PROMPT *  END OF CREATION SCRIPT  *
PROMPT *******************************
--*************************************************************************************
--*                    EOF.                        *
--*************************************************************************************
```

2- SC_USERS.SQL

```
--*****************************************************
--*              CREATION SCRIPT
--*                   FOR USERS
--*****************************************************

CONNECT SYS/*****@***** AS SYSDBA;

--***********************************************************
--*        USER DESTRUCTION
--***********************************************************

DROP USER USR_CDN;
DROP USER USR_COMMUN;



--***********************************************************
--*        USERS CREATION
--***********************************************************

CREATE USER USR_COMMUN IDENTIFIED BY COMMUN
          DEFAULT TABLESPACE TS_COMMUN3
          TEMPORARY TABLESPACE TS_TEMP3;

CREATE USER USR_CDN IDENTIFIED BY CDN
          DEFAULT TABLESPACE TS_CDN
          TEMPORARY TABLESPACE TS_TEMP3;


PROMPT *******************************
PROMPT *  END OF CREATION SCRIPT  *
PROMPT *******************************
--*************************************************************************************
--*                    EOF.                        *
--*************************************************************************************
```

3- SC_TA_TYPE_DEF.SQL

```
--*****************************************************
--*          CREATION SCRIPT FOR
--*          TABLE USR_COMMUN.TYPE_DEFINITION
--*
--*****************************************************


--*********  CONNECTION BY THE DBA
CONNECT SYS/*******@*** AS SYSDBA;

--******** GRANTS TO THE TABLE CREATOR AND CONNECTION
GRANT RESOURCE, DBA ,CREATE TABLE TO USR_COMMUN;

CONNECT USR_COMMUN/COMMUN@***;

--***********************************************************
--*        DESTRUCTION DES TABLES
--***********************************************************

DROP TABLE USR_COMMUN.TYPE_DEFINITION;

--*****************************************************
--*          TABLE CREATION
--*****************************************************

CREATE TABLE USR_COMMUN.TYPE_DEFINITION(
          TYPE_ID                       NUMBER(3) NOT NULL,
          TYPE_NAME                     VARCHAR2(25) NOT NULL,
          TYPE_NOM                      VARCHAR2(25) NOT NULL,
          TYPE_DESCRIPTION_EN           VARCHAR2(250) NOT NULL,
          TYPE_DESCRIPTION_FR           VARCHAR2(250) NOT NULL,
          CONSTRAINT TYPE_DEF_PK PRIMARY KEY (TYPE_ID))
          TABLESPACE TS_COMMUN3;


--**** DROITS DE REFERENCE A USR_CDN
GRANT REFERENCES ON USR_COMMUN.TYPE_DEFINITION TO USR_CDN;


-- *******   INSERTION OF INFORMATION ABOUT ALREADY KNOWN DATA TYPE

INSERT INTO USR_COMMUN.TYPE_DEFINITION VALUES(
          200, 'SHORELINE DATA TYPE', ' TYPE DE DONNEES LIGNE DE RIVAGE ', 'SHORELINE
DATA TYPE INCLUDING A SPATIAL OBJECT FOR LINE DESCRIPTION', 'LIGNE DE RIVAGE
INCLUANT DES OBJETS SPATIAUX DE TYPE LIGNE');

INSERT INTO USR_COMMUN.TYPE_DEFINITION VALUES(
          201, 'POLITICAL UNIT DATA TYPE', 'TYPE DE DONNEES UNITEE POLITIQUE ',
'POLITICAL UNIT DATA TYPE INCLUDING A SPATIAL OBJECT FOR BOUNDARY
DESCRIPTION', 'UNITEE POLITIQUE INCLUANT DES OBJETS SPATIAUX POUR LA
DELIMITATION DES FRONTIERES');

INSERT INTO USR_COMMUN.TYPE_DEFINITION VALUES(
```

```
        202, 'TRIBUTARY DATA TYPE', ‘TYPE DE DONNEES TRIBUTAIRE’, 'TRIBUTARY DATA TYPE INCLUDING A SPATIAL OBJECT TO DESCRIBE A TRIBUTARY', ‘TYPE DE DONNEES DECRIVANT LES TRIBUTAIRES INCLUANT DES OBJETS SPATIAUX');

INSERT INTO USR_COMMUN.TYPE_DEFINITION VALUES(
        203, 'LIDAR DATA TYPE', ‘TYPE DE DONNEES LIDAR’, 'DIGITAL ELEVATION MODEL (LIDAR) DATA TYPE INCLUDING A SPATIAL OBJECT FOR POINT DESCRIPTION', ‘DONNEES DE TYPE MODELE DE TERRAIN (LIDAR) INCLUANT DES OBJETS SPATIAUX DE TYPE POINT');

INSERT INTO USR_COMMUN.TYPE_DEFINITION VALUES(
        204, 'GRID DATA TYPE', ‘TYPE DE DONNÉES ELEVATION SUR GRID’, 'REGULAR 5 METER GRID WITH AVERAGE HEIGHT INCLUDING A SPATIAL OBJECT FOR POINT DESCRIPTION', ‘GRID REGULIER DE 5 METRE AVEC HAUTEUR MOYENNE INCLUANT DES OBJETS SPATIAUX DE TYPE POINT');

INSERT INTO USR_COMMUN.TYPE_DEFINITION VALUES(
        205, 'BATHMETRY DATA TYPE', ‘TYPE DE DONNEES BATHYMETRIE’, 'BATHYMETRIC DATA INCLUDING A SPATIAL OBJECT FOR POINT DESCRIPTION', ‘TYPE DE DONNEES BATHYMETRIE INCLUANT DES OBJETS SPATIAUX DE TYPE POINT');

INSERT INTO USR_COMMUN.TYPE_DEFINITION VALUES(
        206, 'CONTOUR DATA TYPE', TYPE DE DONNEES CONTOUR ELEVATION’, 'FOUR METER INTERVAL CONTOUR LINE DESCRIBED AS SPATIAL OBJECT TO REPRESENT THE LINES', ‘CONTOUR A INTERVAL DE QUATRE METRES INCLUANT DES OBJETS SPATIAUX DE TYPE LIGNE');

INSERT INTO USR_COMMUN.TYPE_DEFINITION VALUES(
        207, 'CMU DATA TYPE', ‘TYPE DE DONNEES UNITE DE CONSERVATION’, 'CONSERVATION MANAGEMENT UNIT DATA TYPE INCLUDING A SPATIAL OBJECT TO DESCRIBE EACH UNIT', ‘TYPE DE DONNEES DECRIVANT DES UNITES DE CONSERVATION FAUNIQUE INCLUANT DES OBJETS SPATIAUX DE TYPE POLYGONE);



-- RECONNECTION BY THE DBS
CONNECT SYS/********@**** AS SYSDBA;

--*************************************************************
--*         RIGHTS REVOCATION FROM THE CREATOR
--*************************************************************

REVOKE RESOURCE, DBA, CREATE TABLE  FROM USR_COMMUN;

-- WE ACCEPT THE TRANSACTIONS
COMMIT;

PROMPT ********************************
PROMPT *  LE SCRIPT SQL EST TERMINÉ  *
PROMPT ********************************

--*****************************************************************************
--*                    FIN DU FICHIER SQL.                    *
--*****************************************************************************
```

4- SC_TA_METADATA_CATALOG.SQL

```
--*****************************************************
--*          CREATION SCRIPT FOR
--*          TABLE USR_COMMUN.METADATA_CATALOG
--*****************************************************

--*********  CONNECTION BY THE DBA
CONNECT SYS/****@**** AS SYSDBA;

--******** GRANTS TO THE TABLE CREATOR AND CONNECTION
GRANT RESOURCE, DBA ,CREATE TABLE TO USR_COMMUN;

CONNECT USR_COMMUN/COMMUN@****;

--*********************************************************
--*         DESTRUCTION DES TABLES
--*********************************************************

DROP TABLE USR_COMMUN.METADATA_CATALOG;

--*****************************************************
--*          TABLE CREATION
--*****************************************************

CREATE TABLE USR_COMMUN.METADATA_CATALOG(
          DATS_ID                       NUMBER(3) NOT NULL,
          XML_ADRESS                    VARCHAR2(100) NOT NULL,
          CONSTRAINT META_CAT_PK PRIMARY KEY (DATS_ID))
          TABLESPACE TS_COMMUN3;

--**** DROITS DE REFERENCE A USR_CDN
GRANT REFERENCES ON USR_COMMUN.METADATA_CATALOG TO USR_CDN;

-- RECONNECTION BY THE DBS
CONNECT SYS/********@**** AS SYSDBA;

--*********************************************************
--*         RIGHTS REVOCATION FROM THE CREATOR
--*********************************************************

REVOKE RESOURCE, DBA, CREATE TABLE  FROM USR_COMMUN;


-- WE ACCEPT THE TRANSACTIONS
COMMIT;

PROMPT ********************************
PROMPT *  LE SCRIPT SQL EST TERMINÉ  *
PROMPT ********************************

--*************************************************************************
--*                    FIN DU FICHIER SQL.                    *
--*************************************************************************
```

5- SC_TA_SHORELINE.SQL

```
--*****************************************************
--*          CREATION SCRIPT FOR
--*          TABLE USR_CDN.SHORELINE
--*
--*****************************************************

--*********  CONNECTION BY THE DBA
CONNECT SYS/*****@*** AS SYSDBA;

--******** GRANTS TO THE TABLE CREATOR AND CONNECTION
GRANT RESOURCE, DBA ,CREATE TABLE TO USR_CDN;

CONNECT USR_CDN/CDN@****;

--***********************************************************
--*         INDEXES DROPPING
--***********************************************************

DROP INDEX USR_CDN.SHORELINE_SX;

--***********************************************************
--*         DESTRUCTION OF VIEW AND TABLE
--***********************************************************

DROP VIEW USR_CDN.V_SHORELINE;
DROP TABLE USR_CDN.SHORELINE;

--** WE REMOVE THE LINE IN THE MDSYS.USER_SDO_GEOM_METADATA VIEW
--** THAT CORRESPONDS TO THE SPATIAL TABLE/OBJECT PAIR
DELETE FROM MDSYS.USER_SDO_GEOM_METADATA WHERE TABLE_NAME =
'SHORELINE';


-- WE DROP THE SEQUENCE ASSOCIATED WITH THE OBJECTS
DROP SEQUENCE USR_CDN.SHORELINE_SEQ;


--*****************************************************
--*          TABLE CREATION
--*****************************************************

CREATE TABLE USR_CDN.SHORELINE(
          SHORELINE_ID                                    NUMBER(10) NOT NULL,
          NAME                                            VARCHAR2(50),
          LINE                                            MDSYS.SDO_GEOMETRY    NOT
NULL,
    TYPE_ID                    NUMBER(10)  NOT NULL,
    DATS_ID                    NUMBER(10)  NOT NULL,
          CONSTRAINT SHORELINE_UNIQUE UNIQUE(SHORELINE_ID),
          CONSTRAINT    FK_TYPE_DEFINITION_SHORELINE    FOREIGN    KEY    (TYPE_ID)
REFERENCES USR_COMMUN.TYPE_DEFINITION(TYPE_ID),
          CONSTRAINT FK_META_SHORELINE FOREIGN KEY (DATS_ID) REFERENCES
                    USR_COMMUN.METADATA_CATALOG(DATS_ID)
          ) TABLESPACE TS_CDN;
```

```
CREATE VIEW USR_CDN.V_SHORELINE AS
        SELECT SHORELINE_ID, NAME, LINE
                FROM USR_CDN.SHORELINE;


-- *******   UPDATING USER_SDO_GEOM_METADATA VIEW
INSERT INTO MDSYS.USER_SDO_GEOM_METADATA VALUES (
'SHORELINE',                    --TABLE NAME
 'LINE',                        -- SPATIAL OBJECT NAME
        MDSYS.SDO_DIM_ARRAY(    -- INFORMATION ABOUT DIMENSIONS
        MDSYS.SDO_DIM_ELEMENT('X', 5000000, 8100000, 0.001),
        MDSYS.SDO_DIM_ELEMENT('Y', -480000, 1870000 , 0.001) ),
        3000000 -- INFORMATION ON COORDINATE SYSTEM (LAMBERT CONFORMAL
CONIC)
         );

-- ********************   INDEXES CREATION
--*** A SPATIAL INDEX
--* THIS INDEX HAS TO BE REBUILT AFTER DATA INSERTION

CREATE INDEX USR_CDN.SHORELINE_SX ON USR_CDN.SHORELINE(LINE)
        INDEXTYPE IS MDSYS.SPATIAL_INDEX
        PARAMETERS ('TABLESPACE=TS_INDEX3');

-- ***  SEQUENCE CREATION FOR SHORELINE IDENTIFICATION
CREATE SEQUENCE USR_CDN.SHORELINE_SEQ
    INCREMENT BY 1
    START WITH 1
    NOMAXVALUE
    NOCYCLE
    CACHE 100;

-- RECONNECTION BY THE DBS
CONNECT SYS/********@**** AS SYSDBA;

--************************************************************
--*         RIGHTS REVOCATION FROM THE CREATOR
--************************************************************

REVOKE RESOURCE, DBA, CREATE TABLE  FROM USR_CDN;

-- WE ACCEPT THE TRANSACTIONS
COMMIT;

PROMPT *******************************
PROMPT *  LE SCRIPT SQL EST TERMINÉ  *
PROMPT *******************************

--*****************************************************************************
--*                     FIN DU FICHIER SQL.                       *
--*****************************************************************************
```

6- SC_TA_POLITICAL_UNIT.SQL

```
--*******************************************************
--*          CREATION SCRIPT FOR
--*          TABLE USR_CDN.POLITICAL_UNIT
--*  FOR STORAGE OF MUNICIPAL, COUNTY, PROVINCIAL
--* AND FEDERAL BOUNDARIES
--*******************************************************

--*********  CONNECTION BY THE DBA
CONNECT SYS/*******@***** AS SYSDBA;

--******** GRANTS TO THE TABLE CREATOR AND CONNECTION
GRANT RESOURCE, DBA ,CREATE TABLE TO USR_CDN;

CONNECT USR_CDN/CDN@****;

--*********************************************************
--*         INDEXES DROPPING
--*********************************************************

DROP INDEX USR_CDN.BOUNDARY_SX;

--*********************************************************
--*         DESTRUCTION OF TABLE AND VIEW
--*********************************************************

DROP VIEW USR_CDN.V_POL_UNIT;
DROP TABLE USR_CDN.POLITICAL_UNIT;

--** WE REMOVE THE LINE IN THE MDSYS.USER_SDO_GEOM_METADATA VIEW
--** THAT CORRESPONDS TO THE SPATIAL TABLE/OBJECT PAIR
DELETE FROM MDSYS.USER_SDO_GEOM_METADATA WHERE TABLE_NAME =
'POLITICAL_UNIT';

-- WE DROP THE SEQUENCE ASSOCIATED WITH THE OBJECTS
DROP SEQUENCE USR_CDN.POL_UNIT_SEQ;

--*******************************************************
--*          TABLE AND VIEW CREATION
--*******************************************************

CREATE TABLE USR_CDN.POLITICAL_UNIT(
          UNIT_ID                                          NUMBER(10) NOT NULL,
          UNIT_NAME                                        VARCHAR2(50),
          BOUNDARY                                         MDSYS.SDO_GEOMETRY   NOT
NULL,
    TYPE_EN                VARCHAR2(20)  CHECK(UPPER(TYPE_EN)  IN  ('MUNICIPAL',
'COUNTY', 'STATE', 'PROVINCIAL', 'FEDERAL')),
    TYPE_FR                VARCHAR2(20)  CHECK(UPPER(TYPE_FR)  IN  ('MUNICIPAL',
'CONTE', 'ETAT', 'PROVINCIAL', 'FEDERALE')),
          TYPE_ID                                          NUMBER(10) NOT NULL,
    DATS_id                NUMBER(10)  NOT NULL,
          CONSTRAINT POL_UNIT_UNIQUE UNIQUE (UNIT_ID),
```

```
	CONSTRAINT FK_TYPE_DEFINITION_POL_UNIT FOREIGN KEY (TYPE_ID)
REFERENCES USR_COMMUN.TYPE_DEFINITION(TYPE_ID),
	CONSTRAINT FK_META_POL_UNITS FOREIGN KEY (DATS_ID) REFERENCES
		USR_COMMUN.METADATA_CATALOG(DATS_ID)
	) TABLESPACE TS_CDN;

CREATE VIEW USR_CDN.V_POL_UNIT AS
	SELECT UNIT_ID, UNIT_NAME, BOUNDARY, TYPE_EN, TYPE_FR
		FROM USR_CDN.POLITICAL_UNIT;

-- *******   UPDATING USER_SDO_GEOM_METADATA VIEW
INSERT INTO MDSYS.USER_SDO_GEOM_METADATA VALUES (
'POLITICAL_UNIT',                       --TABLE NAME
 'BOUNDARY',                            -- SPATIAL OBJECT NAME
	MDSYS.SDO_DIM_ARRAY(      -- INFORMATION ABOUT DIMENSIONS
	MDSYS.SDO_DIM_ELEMENT('X', 5000000, 8100000, 0.001),
	MDSYS.SDO_DIM_ELEMENT('Y', -480000, 1870000 , 0.001) ),
	3000000 -- INFORMATION ON COORDINATE SYSTEM (LAMBERT CONFORMAL
CONIC)
	 );

-- **********************   INDEXES CREATION
--*** A SPATIAL INDEX
--* THIS INDEX HAS TO BE REBUILT AFTER DATA INSERTION FOR BEST RESULTS

CREATE INDEX USR_CDN.BOUNDARY_SX ON USR_CDN.POLITICAL_UNIT(BOUNDARY)
	INDEXTYPE IS MDSYS.SPATIAL_INDEX
	PARAMETERS ('TABLESPACE=TS_INDEX3');

-- ***  SEQUENCE CREATION FOR SHORELINE IDENTIFICATION
CREATE SEQUENCE USR_CDN.POL_UNIT_SEQ
    INCREMENT BY 1
    START WITH 1
    NOMAXVALUE
    NOCYCLE
    CACHE 100;

-- RECONNECTION BY THE DBA
CONNECT SYS/********@**** AS SYSDBA;

--************************************************************
--*         RIGHTS REVOCATION FROM THE CREATOR
--************************************************************

REVOKE RESOURCE, DBA, CREATE TABLE  FROM USR_CDN;

-- WE ACCEPT THE TRANSACTIONS
COMMIT;

PROMPT *******************************
PROMPT *  LE SCRIPT SQL EST TERMINÉ  *
PROMPT *******************************

--*****************************************************************************
--*                     FIN DU FICHIER SQL.                       *
--*****************************************************************************
```

7- SC_TA_TRIBUTARY.SQL

```
--*******************************************************
--*          CREATION SCRIPT FOR
--*          TABLE USR_CDN.TRIBUTARY
--*         FOR TRIBUTARIES STORAGE
--*******************************************************

--*********  CONNECTION BY THE DBA
CONNECT SYS/*******@**** AS SYSDBA;

--******** GRANTS TO THE TABLE CREATOR AND CONNECTION
GRANT RESOURCE, DBA ,CREATE TABLE TO USR_CDN;

CONNECT USR_CDN/CDN@****;

--*************************************************************
--*         INDEXES DROPPING
--*************************************************************

DROP INDEX USR_CDN.TRIBUTARY_SX;

--*************************************************************
--*         DESTRUCTION OF VIEW AND TABLE
--*************************************************************

DROP VIEW USR_CDN.V_TRIBUTARY;
DROP TABLE USR_CDN.TRIBUTARY;

--** WE REMOVE THE LINE IN THE MDSYS.USER_SDO_GEOM_METADATA VIEW
--** THAT CORRESPONDS TO THE SPATIAL TABLE/OBJECT PAIR
DELETE FROM MDSYS.USER_SDO_GEOM_METADATA WHERE TABLE_NAME =
'TRIBUTARY';


-- WE DROP THE SEQUENCE ASSOCIATED WITH THE OBJECTS

DROP SEQUENCE USR_CDN.TRIBUTARY_SEQ;

--*******************************************************
--*          TABLE AND VIEW CREATION
--*******************************************************

CREATE TABLE USR_CDN.TRIBUTARY(
          TRIBUTARY_ID                                    NUMBER(10) NOT NULL,
          WATERSHED_ID                                    NUMBER(10) NOT NULL,
          NAME                                            VARCHAR2(50),
          TRIBUTARY                                       MDSYS.SDO_GEOMETRY    NOT
NULL,
     TYPE_ID                    NUMBER(10)  NOT NULL,
     DATS_ID                    NUMBER(10)  NOT NULL,
          CONSTRAINT TRIBUTARY_UNIQUE UNIQUE(TRIBUTARY_ID),
          CONSTRAINT    FK_TYPE_DEFINITION_TRIBUTARY    FOREIGN    KEY    (TYPE_ID)
REFERENCES USR_COMMUN.TYPE_DEFINITION(TYPE_ID),
          CONSTRAINT FK_META_TRIBUTARY FOREIGN KEY (DATS_ID) REFERENCES
```

```
                USR_COMMUN.METADATA_CATALOG(DATS_ID)
        ) TABLESPACE TS_CDN;

CREATE VIEW USR_CDN.V_TRIBUTARY AS
        SELECT TRIBUTARY_ID, WATERSHED_ID, NAME, TRIBUTARY
                FROM USR_CDN.TRIBUTARY;

-- *******   UPDATING USER_SDO_GEOM_METADATA VIEW
INSERT INTO MDSYS.USER_SDO_GEOM_METADATA VALUES (
'TRIBUTARY',                    --TABLE NAME
 'TRIBUTARY',                           -- SPATIAL OBJECT NAME
        MDSYS.SDO_DIM_ARRAY(    -- INFORMATION ABOUT DIMENSIONS
        MDSYS.SDO_DIM_ELEMENT('X', 5000000, 8100000, 0.001),
        MDSYS.SDO_DIM_ELEMENT('Y', -480000, 1870000 , 0.001) ),
        3000000 -- INFORMATION ON COORDINATE SYSTEM (LAMBERT CONFORMAL
CONIC)
         );

-- ********************   INDEXES CREATION
--*** A SPATIAL INDEX
--* THIS INDEX HAS TO BE REBUILT AFTER DATA INSERTION

CREATE INDEX USR_CDN.TRIBUTARY_SX ON USR_CDN.TRIBUTARY(TRIBUTARY)
        INDEXTYPE IS MDSYS.SPATIAL_INDEX
        PARAMETERS ('TABLESPACE=TS_INDEX3');

-- ***  SEQUENCE CREATION FOR SHORELINE IDENTIFICATION
CREATE SEQUENCE USR_CDN.TRIBUTARY_SEQ
    INCREMENT BY 1
    START WITH 1
    NOMAXVALUE
    NOCYCLE
    CACHE 100;

COMMIT;

-- RECONNECTION BY THE DBS
CONNECT SYS/********@**** AS SYSDBA;

--************************************************************
--*         RIGHTS REVOCATION FROM THE CREATOR
--************************************************************

REVOKE RESOURCE, DBA, CREATE TABLE  FROM USR_CDN;

-- WE ACCEPT THE TRANSACTIONS
COMMIT;

PROMPT *******************************
PROMPT *  LE SCRIPT SQL EST TERMINÉ  *
PROMPT *******************************

--*****************************************************************************
--*                     FIN DU FICHIER SQL.                       *
--*****************************************************************************
```

8- SC_TA_DEM.SQL

```
--*******************************************************
--*          CREATION SCRIPT FOR
--*          TABLE USR_CDN.DEM
--*         FOR DIGITAL ELEVATION MODEL
--*******************************************************


--*********  CONNECTION BY THE DBA
CONNECT SYS/******@**** AS SYSDBA;

--******** GRANTS TO THE TABLE CREATOR AND CONNECTION
GRANT RESOURCE, DBA ,CREATE TABLE TO USR_CDN;

CONNECT USR_CDN/CDN@****;

--*************************************************************
--*        INDEXES DROPPING
--*************************************************************

DROP INDEX USR_CDN.LIDAR_SX;
DROP INDEX USR_CDN.GRID_SX;
DROP INDEX USR_CDN.BATHYMETRY_SX;
DROP INDEX USR_CDN.CONTOUR_SX;

--*************************************************************
--*        DESTRUCTION OF TABLE AND VIEW
--*************************************************************

DROP VIEW USR_CDN.V_LIDAR;
DROP TABLE USR_CDN.LIDAR;

DROP VIEW USR_CDN.V_GRID;
DROP TABLE USR_CDN.GRID;

DROP VIEW USR_CDN.V_BATHYMETRY;
DROP TABLE USR_CDN.BATHYMETRY;

DROP VIEW USR_CDN.V_CONTOUR;
DROP TABLE USR_CDN.CONTOUR;


--** WE REMOVE THE LINE IN THE MDSYS.USER_SDO_GEOM_METADATA VIEW
--** THAT CORRESPONDS TO THE SPATIAL TABLE/OBJECT PAIR

DELETE FROM MDSYS.USER_SDO_GEOM_METADATA WHERE TABLE_NAME = 'LIDAR';
DELETE FROM MDSYS.USER_SDO_GEOM_METADATA WHERE TABLE_NAME = 'GRID';
DELETE FROM MDSYS.USER_SDO_GEOM_METADATA WHERE TABLE_NAME =
'BATHYMETRY';
DELETE FROM MDSYS.USER_SDO_GEOM_METADATA WHERE TABLE_NAME = 'CONTOUR';
```

```
-- WE DROP THE SEQUENCE ASSOCIATED WITH THE OBJECTS

DROP SEQUENCE USR_CDN.LIDAR_SEQ;
DROP SEQUENCE USR_CDN.GRID_SEQ;
DROP SEQUENCE USR_CDN.BATHYMETRY_SEQ;
DROP SEQUENCE USR_CDN.CONTOUR_SEQ;


--*****************************************************
--*          TABLE AND VIEW CREATION
--*****************************************************

CREATE TABLE USR_CDN.LIDAR(
        POINT_ID                        NUMBER(10) NOT NULL,
        POINT                           MDSYS.SDO_GEOMETRY NOT NULL,
    TYPE_ID                         NUMBER(10)  NOT NULL,
    DATS_ID                         NUMBER(10)  NOT NULL,
        CONSTRAINT FK_TYPE_DEFINITION_LIDAR FOREIGN KEY (TYPE_ID) REFERENCES
USR_COMMUN.TYPE_DEFINITION(TYPE_ID),
        CONSTRAINT LIDAR_PK PRIMARY KEY (POINT_ID),
        CONSTRAINT FK_META_CAT_LIDAR FOREIGN KEY (DATS_ID) REFERENCES
                USR_COMMUN.METADATA_CATALOG(DATS_ID))
        ORGANIZATION INDEX TABLESPACE TS_CDN
        PCTTHRESHOLD 20
        OVERFLOW TABLESPACE TS_CDN
        NOCOMPRESS;

CREATE TABLE USR_CDN.BATHYMETRY(
        POINT_ID                        NUMBER(10) NOT NULL,
        POINT                           MDSYS.SDO_GEOMETRY NOT NULL,
    TYPE_ID                         NUMBER(10)  NOT NULL,
    DATS_ID                         NUMBER(10)  NOT NULL,
        CONSTRAINT FK_TYPE_DEFINITION_BATHYMETRY FOREIGN KEY (TYPE_ID)
REFERENCES USR_COMMUN.TYPE_DEFINITION(TYPE_ID),
        CONSTRAINT BATHY_PK PRIMARY KEY (POINT_ID),
        CONSTRAINT FK_META_CAT_BATHYMETRY FOREIGN KEY (DATS_ID) REFERENCES
                USR_COMMUN.METADATA_CATALOG(DATS_ID))
        ORGANIZATION INDEX TABLESPACE TS_CDN
        PCTTHRESHOLD 20
        OVERFLOW TABLESPACE TS_CDN
        NOCOMPRESS;

CREATE TABLE USR_CDN.GRID(
        POINT_ID                        NUMBER(10) NOT NULL,
        POINT                           MDSYS.SDO_GEOMETRY NOT NULL,
        FLAG                            NUMBER(10, 4),
    TYPE_ID                         NUMBER(10)  NOT NULL,
    DATS_ID                         NUMBER(10)  NOT NULL,
        CONSTRAINT FK_TYPE_DEFINITION_GRID FOREIGN KEY (TYPE_ID) REFERENCES
USR_COMMUN.TYPE_DEFINITION(TYPE_ID),
        CONSTRAINT GRID_PK PRIMARY KEY (POINT_ID),
        CONSTRAINT FK_META_CAT_GRID FOREIGN KEY (DATS_ID) REFERENCES
                USR_COMMUN.METADATA_CATALOG(DATS_ID))
        ORGANIZATION INDEX TABLESPACE TS_CDN
        PCTTHRESHOLD 20
        OVERFLOW TABLESPACE TS_CDN
```

```
        NOCOMPRESS;

CREATE TABLE USR_CDN.CONTOUR(
        LINE_ID                                 NUMBER(10) NOT NULL,
        LINE                                    MDSYS.SDO_GEOMETRY NOT NULL,
        ELEVATION                                       NUMBER(16, 4),
    TYPE_ID                             NUMBER(10)  NOT NULL,
    DATS_ID                             NUMBER(10)  NOT NULL,
        CONSTRAINT      FK_TYPE_DEFINITION_CONTOUR      FOREIGN     KEY     (TYPE_ID)
REFERENCES USR_COMMUN.TYPE_DEFINITION(TYPE_ID),
        CONSTRAINT CONTOUR_PK PRIMARY KEY (LINE_ID),
        CONSTRAINT FK_META_CAT_CONTOUR FOREIGN KEY (DATS_ID) REFERENCES
                USR_COMMUN.METADATA_CATALOG(DATS_ID))
        ORGANIZATION INDEX TABLESPACE TS_CDN
        PCTTHRESHOLD 20
        OVERFLOW TABLESPACE TS_CDN
        NOCOMPRESS;



CREATE VIEW USR_CDN.V_LIDAR AS
        SELECT POINT_ID, POINT
                FROM USR_CDN.LIDAR;

CREATE VIEW USR_CDN.V_GRID AS
        SELECT POINT_ID, POINT, FLAG
                FROM USR_CDN.GRID;

CREATE VIEW USR_CDN.V_BATHYMETRY AS
        SELECT POINT_ID, POINT
                FROM USR_CDN.BATHYMETRY;

CREATE VIEW USR_CDN.V_CONTOUR AS
        SELECT LINE_ID, LINE, ELEVATION
                FROM USR_CDN.CONTOUR;


-- *******   UPDATING USER_SDO_GEOM_METADATA VIEW
INSERT INTO MDSYS.USER_SDO_GEOM_METADATA VALUES (
'LIDAR',                                --TABLE NAME
 'POINT',                                       -- SPATIAL OBJECT NAME
        MDSYS.SDO_DIM_ARRAY(      -- INFORMATION ABOUT DIMENSIONS
        MDSYS.SDO_DIM_ELEMENT('X', 5000000, 8100000, 0.001),
        MDSYS.SDO_DIM_ELEMENT('Y', -480000, 1870000 , 0.001) ),
        3000000 -- INFORMATION ON COORDINATE SYSTEM (LAMBERT CONFORMAL
CONIC)
         );

INSERT INTO MDSYS.USER_SDO_GEOM_METADATA VALUES (
'GRID',                                 --TABLE NAME
 'POINT',                                       -- SPATIAL OBJECT NAME
        MDSYS.SDO_DIM_ARRAY(      -- INFORMATION ABOUT DIMENSIONS
        MDSYS.SDO_DIM_ELEMENT('X', 5000000, 8100000, 0.001),
        MDSYS.SDO_DIM_ELEMENT('Y', -480000, 1870000 , 0.001) ),
```

```
        3000000 -- INFORMATION ON COORDINATE SYSTEM (LAMBERT CONFORMAL
CONIC)
         );

INSERT INTO MDSYS.USER_SDO_GEOM_METADATA VALUES (
'BATHYMETRY',                          --TABLE NAME
 'POINT',                              -- SPATIAL OBJECT NAME
        MDSYS.SDO_DIM_ARRAY(      -- INFORMATION ABOUT DIMENSIONS
        MDSYS.SDO_DIM_ELEMENT('X', 5000000, 8100000, 0.001),
        MDSYS.SDO_DIM_ELEMENT('Y', -480000, 1870000 , 0.001) ),
        3000000 -- INFORMATION ON COORDINATE SYSTEM (LAMBERT CONFORMAL
CONIC)
         );

INSERT INTO MDSYS.USER_SDO_GEOM_METADATA VALUES (
'CONTOUR',                      --TABLE NAME
 'LINE',                        -- SPATIAL OBJECT NAME
        MDSYS.SDO_DIM_ARRAY(      -- INFORMATION ABOUT DIMENSIONS
        MDSYS.SDO_DIM_ELEMENT('X', 5000000, 8100000, 0.001),
        MDSYS.SDO_DIM_ELEMENT('Y', -480000, 1870000 , 0.001) ),
        3000000 -- INFORMATION ON COORDINATE SYSTEM (LAMBERT CONFORMAL
CONIC)
         );



-- ********************   INDEXES CREATION
--*** FOUR SPATIAL INDEXES
--* THESE INDEX HAS TO BE REBUILT AFTER DATA INSERTION FOR BEST RESULTS

CREATE INDEX USR_CDN.LIDAR_SX ON USR_CDN.LIDAR(POINT)
        INDEXTYPE IS MDSYS.SPATIAL_INDEX
        PARAMETERS ('TABLESPACE=TS_INDEX3');

CREATE INDEX USR_CDN.GRID_SX ON USR_CDN.GRID(POINT)
        INDEXTYPE IS MDSYS.SPATIAL_INDEX
        PARAMETERS ('TABLESPACE=TS_INDEX3');

CREATE INDEX USR_CDN.BATHYMETRY_SX ON USR_CDN.BATHYMETRY(POINT)
        INDEXTYPE IS MDSYS.SPATIAL_INDEX
        PARAMETERS ('TABLESPACE=TS_INDEX3');

CREATE INDEX USR_CDN.CONTOUR_SX ON USR_CDN.CONTOUR(LINE)
        INDEXTYPE IS MDSYS.SPATIAL_INDEX
        PARAMETERS ('TABLESPACE=TS_INDEX3');



-- ***  SEQUENCE CREATION FOR SHORELINE IDENTIFICATION
CREATE SEQUENCE USR_CDN.LIDAR_SEQ
    INCREMENT BY 1
    START WITH 1
    NOMAXVALUE
    NOCYCLE
    CACHE 100;
```

```
CREATE SEQUENCE USR_CDN.GRID_SEQ
    INCREMENT BY 1
    START WITH 1
    NOMAXVALUE
    NOCYCLE
    CACHE 100;

CREATE SEQUENCE USR_CDN.BATHYMETRY_SEQ
    INCREMENT BY 1
    START WITH 1
    NOMAXVALUE
    NOCYCLE
    CACHE 100;

CREATE SEQUENCE USR_CDN.CONTOUR_SEQ
    INCREMENT BY 1
    START WITH 1
    NOMAXVALUE
    NOCYCLE
    CACHE 100;



-- RECONNECTION BY THE DBA
CONNECT SYS/********@**** AS SYSDBA;

--*************************************************************
--*         RIGHTS REVOCATION FROM THE CREATOR
--*************************************************************

REVOKE RESOURCE, DBA, CREATE TABLE  FROM USR_CDN;

-- WE ACCEPT THE TRANSACTIONS
COMMIT;

PROMPT *******************************
PROMPT *  LE SCRIPT SQL EST TERMINÉ  *
PROMPT *******************************

--*****************************************************************************
--*                     FIN DU FICHIER SQL.                       *
--*****************************************************************************
```

9- SC_TA_CMU.SQL

```
--*****************************************************
--*          CREATION SCRIPT FOR
--*          TABLE USR_CDN.CMU
--*        FOR CONSERVATION MANAGEMENT UNITS STORAGE
--*****************************************************

--*********  CONNECTION BY THE DBA
CONNECT SYS/********@**** AS SYSDBA;

--******** GRANTS TO THE TABLE CREATOR AND CONNECTION
GRANT RESOURCE, DBA ,CREATE TABLE TO USR_CDN;

CONNECT USR_CDN/CDN@*****;

--***********************************************************
--*        INDEXES DROPPING
--***********************************************************

DROP INDEX USR_CDN.CMU_SX;


--***********************************************************
--*        DESTRUCTION OF TABLE AND VIEW
--***********************************************************
DROP VIEW USR_CDN.V_CMU;
DROP TABLE USR_CDN.CMU;


--** WE REMOVE THE LINE IN THE MDSYS.USER_SDO_GEOM_METADATA VIEW
--** THAT CORRESPONDS TO THE SPATIAL TABLE/OBJECT PAIR

DELETE FROM MDSYS.USER_SDO_GEOM_METADATA WHERE TABLE_NAME = 'CMU';


-- WE DROP THE SEQUENCES ASSOCIATED WITH THE SPATIAL OBJECTS IN THE TABLE
DROP SEQUENCE USR_CDN.UNIT_ID_SEQ;




--*****************************************************
--*          TABLE AND VIEW CREATION
--*****************************************************

CREATE TABLE USR_CDN.CMU(
        UNIT_ID                 NUMBER(10) NOT NULL,
        TYPE                    VARCHAR2(5) NOT NULL,     --        TYPE        OF
CONSERVATION UNIT
        UNIT_NAME               VARCHAR2(75) NOT NULL,    -- OFFICIAL NAME
        DESIG                   VARCHAR2(75) NOT NULL,    --        REASON      FOR
CONSERVATION
```

```
        CMU                             MDSYS.SDO_GEOMETRY NOT NULL,-- SPATIAL OBJET
FOR POLYGONE
        CAT_IUCN                        VARCHAR2(20) NOT NULL,      --                  UICN
CLASSIFICATION
        CREATION                        DATE,
    TYPE_ID                     NUMBER(10)  NOT NULL,
    DATS_ID                     NUMBER(10)  NOT NULL,
        CONSTRAINT CAT_IUCN_CST CHECK (UPPER(CAT_IUCN) IN ('IA', 'IB', 'II', 'III', 'IV', 'V',
'VI')),
        CONSTRAINT PK_CMU PRIMARY KEY(UNIT_ID),
        CONSTRAINT FK_TYPE_DEFINITION_CMU FOREIGN KEY (TYPE_ID) REFERENCES
                USR_COMMUN.TYPE_DEFINITION(TYPE_ID),
        CONSTRAINT FK_META_CAT_CMU FOREIGN KEY (DATS_ID) REFERENCES
                USR_COMMUN.METADATA_CATALOG(DATS_ID)
        ) TABLESPACE TS_CDN;

CREATE OR REPLACE VIEW USR_CDN.V_CMU AS
   SELECT UNIT_ID, TYPE, UNIT_NAME, DESIG, CMU, CAT_IUCN, CREATION
   FROM USR_CDN.CMU;


-- *******   UPDATING USER_SDO_GEOM_METADATA VIEW
-- ** ONE ROW MUST BE ADDED TO THE MDSYS.USER_SDO_GEOM_METADATA VIEW

 INSERT INTO MDSYS.USER_SDO_GEOM_METADATA VALUES (
        'CMU',                      --TABLE NAME
        'CMU',              -- SPATIAL OBJECT NAME
        MDSYS.SDO_DIM_ARRAY(      -- INFORMATION ABOUT DIMENSIONS
        MDSYS.SDO_DIM_ELEMENT('X', 5000000, 8100000, 0.001),
        MDSYS.SDO_DIM_ELEMENT('Y', -480000, 1870000 , 0.001) ),
        3000000 -- INFORMATION ON COORDINATE SYSTEM (LAMBERT CONFORMAL
CONIC)
         );

-- **********************   INDEXES CREATION
--*** A SPATIAL INDEX
--* THIS INDEX HAS TO BE REBUILT AFTER DATA INSERTION FOR BEST RESULTS
CREATE INDEX USR_CDN.CMU_SX ON USR_CDN.CMU(CMU)
        INDEXTYPE IS MDSYS.SPATIAL_INDEX
        PARAMETERS ('tablespace=TS_INDEX3');



-- ***  SEQUENCE CREATION FOR CMU SPATIAL OBJECTS IDENTIFICATION
CREATE SEQUENCE USR_CDN.UNIT_ID_SEQ
    INCREMENT BY 1
    START WITH 1
    NOMAXVALUE
    NOCYCLE
    CACHE 100;


-- RECONNECTION BY THE DBS
CONNECT SYS/********@**** AS SYSDBA;

--*********************************************************
```

```
--*         RIGHTS REVOCATION FROM THE CREATOR
--*************************************************************

REVOKE RESOURCE, DBA, CREATE TABLE  FROM USR_CDN;

-- WE ACCEPT THE TRANSACTIONS
COMMIT;

PROMPT ********************************
PROMPT *  LE SCRIPT SQL EST TERMINÉ  *
PROMPT ********************************

--*************************************************************************************
--*                      FIN DU FICHIER SQL.                          *
--*************************************************************************************
```

11- SC_TA_CADASTRAL_INFO.SQL

```
--*****************************************************
--*            CREATION SCRIPT FOR
--*            TABLE USR_CDN.CADAS_INFO
--*  FOR STORAGE OF CADASTRAL INFORMATION
--*
--*****************************************************


--*********  CONNECTION BY THE DBA
CONNECT SYS/PLANTE003D@BD3 AS SYSDBA;

--******** GRANTS TO THE TABLE CREATOR AND CONNECTION
GRANT RESOURCE, DBA ,CREATE TABLE TO USR_CDN;

CONNECT USR_CDN/CDN@BD3;

--*********************************************************
--*          INDEXES DROPPING
--*********************************************************

DROP INDEX USR_CDN.CADAS_POLY_SX;


--*********************************************************
--*          DESTRUCTION OF TABLE AND VIEW
--*********************************************************

DROP VIEW USR_CDN.V_CADAS_POLY;
DROP VIEW USR_CDN.V_CADAS_INFO;

DROP TABLE USR_CDN.CADAS_INFO;
DROP TABLE USR_CDN.CADAS_POLY;



--** WE REMOVE THE LINE IN THE MDSYS.USER_SDO_GEOM_METADATA VIEW
--** THAT CORRESPONDS TO THE SPATIAL TABLE/OBJECT PAIR
DELETE    FROM    MDSYS.USER_SDO_GEOM_METADATA    WHERE    TABLE_NAME    =
'CADAS_POLY';


-- WE DROP THE SEQUENCE ASSOCIATED WITH THE OBJECTS
DROP SEQUENCE USR_CDN.CADAS_POLY_SEQ;
DROP SEQUENCE USR_CDN.CADAS_INFO_SEQ;


--*****************************************************
--*            TABLE AND VIEW CREATION
--*****************************************************

CREATE TABLE USR_CDN.CADAS_POLY(
      CADAS_ID                                    NUMBER(10) NOT NULL,
      POLYGONE                                    MDSYS.SDO_GEOMETRY    NOT
NULL,
```

```
      TYPE_ID                                    NUMBER(10) NOT NULL,
      DATS_ID                                    NUMBER(10)  NOT NULL,
      CONSTRAINT CADAS_POLY_PK PRIMARY KEY(CADAS_ID),
      CONSTRAINT FK_META_CADAS_POLY FOREIGN KEY (DATS_ID) REFERENCES
            USR_COMMUN.METADATA_CATALOG(DATS_ID)
      ) TABLESPACE TS_CDN;


CREATE TABLE USR_CDN.CADAS_INFO(
      DATA_ID                                    NUMBER(10) NOT NULL,
      CADAS_ID                                   NUMBER(10) NOT NULL,
      VALUE                                      NUMBER(23, 16) NOT NULL,
      TYPE_ID                                    NUMBER(10) NOT NULL,
      CONSTRAINT CADAS_INFO_PK PRIMARY KEY(DATA_ID),
      CONSTRAINT FK_TYPE_DEF_CADAS_INFO FOREIGN KEY (TYPE_ID) REFERENCES
USR_COMMUN.TYPE_DEFINITION(TYPE_ID),
      CONSTRAINT  FK_CAD_POLY_CAD_INFO  FOREIGN  KEY(CADAS_ID)  REFERENCES
USR_CDN.CADAS_POLY(CADAS_ID)
      ) TABLESPACE TS_CDN;

CREATE VIEW USR_CDN.V_CADAS_POLY AS
      SELECT CADAS_ID, POLYGONE
            FROM USR_CDN.CADAS_POLY;

CREATE VIEW USR_CDN.V_CADAS_INFO AS
      SELECT DATA_ID, CADAS_ID, VALUE
            FROM USR_CDN.CADAS_INFO;



-- *******   UPDATING USER_SDO_GEOM_METADATA VIEW
INSERT INTO MDSYS.USER_SDO_GEOM_METADATA VALUES (
'CADAS_POLY',                 --TABLE NAME
 'POLYGONE',                        -- SPATIAL OBJECT NAME
      MDSYS.SDO_DIM_ARRAY(    -- INFORMATION ABOUT DIMENSIONS
      MDSYS.SDO_DIM_ELEMENT('X', 5000000, 8100000, 0.001),
      MDSYS.SDO_DIM_ELEMENT('Y', -480000, 1870000 , 0.001) ),
      3000000  --  INFORMATION  ON  COORDINATE  SYSTEM  (LAMBERT  CONFORMAL
CONIC)
       );


-- ********************   INDEXES CREATION
--*** A SPATIAL INDEX
--* THIS INDEX HAS TO BE REBUILT AFTER DATA INSERTION FOR BEST RESULTS

CREATE INDEX USR_CDN.CADAS_POLY_SX ON USR_CDN.CADAS_POLY(POLYGONE)
      INDEXTYPE IS MDSYS.SPATIAL_INDEX
      PARAMETERS ('TABLESPACE=TS_INDEX3');



-- ***  SEQUENCE CREATION FOR CADASTRAL INFORMATION IDENTIFICATION
CREATE SEQUENCE USR_CDN.CADAS_POLY_SEQ
      INCREMENT BY 1
      START WITH 1
      NOMAXVALUE
```

```
      NOCYCLE
      CACHE 100;

CREATE SEQUENCE USR_CDN.CADAS_INFO_SEQ
      INCREMENT BY 1
      START WITH 1
      NOMAXVALUE
      NOCYCLE
      CACHE 100;



-- RECONNECTION BY THE DBA
CONNECT SYS/PLANTE003@BD3 AS SYSDBA;

--*********************************************************
--*          RIGHTS REVOCATION FROM THE CREATOR
--*********************************************************

REVOKE RESOURCE, DBA, CREATE TABLE  FROM USR_CDN;

-- WE ACCEPT THE TRANSACTIONS
COMMIT;

PROMPT ******************************
PROMPT *  LE SCRIPT SQL EST TERMINÉ  *
PROMPT ******************************

--
************************************************************************
******
--*                                                  FIN  DU  FICHIER  SQL.
*
--
************************************************************************
******
```

11- SC_PROJECTION.SQL

```
--*********************************************************
-- SCRIPT DE CREATION D'UNE PROJECTION (LAMBERT CONFORMAL CONIC)
--*********************************************************

-- DELETION DE LA LIGNE
DELETE FROM MDSYS.CS_SRS WHERE SRID = 3000000;

COMMIT;


-- INSERTION D'UNE LIGNE DANS LA TABLE MDSYS.CS_SRS
INSERT INTO mdsys.cs_srs VALUES (
'LAMBERT CONFORMAL CONIC(IJC)/NAD 83',
3000000,
3000000,
'CMI',
'PROJCS["LAMBERT CONFORMAL CONIC",
GEOGCS [ "NAD 83",
DATUM ["NAD 83 ",
SPHEROID ["GRS 80", 6378137.000000, 298.257222]],
PRIMEM [ "Greenwich", 0.000000 ],
UNIT ["Decimal Degree", 0.01745329251994330]],
PROJECTION ["Lambert Conformal Conic"],
PARAMETER ["Scale_Factor", 1.0],
PARAMETER ["Standard_Parallel_1", 49.0],
PARAMETER ["Standard_Parallel_2", 77.0],
PARAMETER ["Central_Meridian", -91.8666],
PARAMETER ["Latitude_of_Origin", -63.0],
PARAMETER ["False_Easting", 6200000.000000],
PARAMETER ["False_Northing", 2958000.000000],
UNIT ["Meter", 1.000000000000]]',
NULL);

COMMIT;

--**** fIN DU SCRIPT
```