Содержание

Введение

В настоящем документе описывается функционал, предусмотренный для загрузки (upload) на Oracle-сервер IMan и выгрузке (download) больших файлов данных, например, данных лазерного сканирования, а также описывается порядок работы при загрузке и выгрузке файлов такого рода.

Алгоритмы работы с данными

Алгоритм загрузки данных

- Установка текущей модели
- Создание карточки файла
  - Создание объекта-карточки файла в модели IMan
  - Создание параметров объекта IMan
- Загрузка файла данных

Алгоритм выгрузки данных

- Установка текущей модели
- Выгрузка файла данных

API IMan работы с данными

Запрос имени текущей модели

Информация о текущей модели доступна из представления SP.V_SYS.

```
Select CURMODEL, CURMODELNAME From SP.V_SYS
```

Поле	**Описание**
CURMODEL	ID текущей модели
CURMODELNAME	Наименование текущей модели

Глобальные параметры пользователя

Для каждого пользователя установлен свой набор значений глобальных параметров, которые доступны из представления SP.V_GLOBALS.

```
select * from SP.V_GLOBALS
```

Среди глобальных параметров также можно найти имя текущей модели, которое задано параметром с именем ***CurModel***.

```
Select * From SP.V_GLOBALS
Where NAME = 'CurModel'
```

Установка текущей модели

Имя текущей задаётся глобальным параметром с именем ***CurModel***.

```
Declare
-- Объект глобального параметра.
p SP.TGPar;
Begin
-- Создаём экземпляр глобального параметра.
-- Перечень глобальных параметров можно найти в SP.V_GLOBAL_PAR_S.
p := SP.TGpar('CurModel');
-- Присваиваем DEFAULT как значение типа строка.
p.VAL := S_('OCKS||Test');
-- Устанавливаем значение. При этом выполняется предопределённая реакция на
-- изменение значения.
p.Save;
End;
```

Создание карточки файла

Создание карточки файла проводится в два этапа. Сначала создается объект модели типа DF.singles.LidarDataFile. Затем созданный объект дополняется необходимым набором параметров. Доступ к объектам модели для чтения и изменения осуществляется через представление SP.V_MODEL_OBJECTS. Доступ к параметрам объектов модели для чтения осуществляется через представление SP.V_MODEL_OBJECT_PARS. Для того чтобы изменить значения параметров объектов модели необходимо воспользоваться объектным типом SP.TMPAR.

Создание объекта-карточки файла

Создание объекта модели осуществляется посредством выполнения запроса вида

```
begin

  Insert Into SP.V_MODEL_OBJECTS
  (PATH,CATALOG_GROUP_NAME, CATALOG_NAME, MOD_OBJ_NAME)
  Values
  ('/Центр им. Д. Рогачёва/Модели/LIDAR_DATA/'
  ,'DF.singles','LidarDataFile','TestLidarData');

  commit;
end;
```

Проверка

Для проверки результатов вставки достаточно выполнить вызов функции, возвращающий ID объекта по его имени.

```
Select SP.MO.MOD_OBJ_ID_BY_FULL_NAME
      ('/Центр им. Д. Рогачёва/Модели/LIDAR_DATA/TestLidarData') As ID
From Dual;
```

Если объект не существует, то запрос

```
Select SP.MO.MOD_OBJ_ID_BY_FULL_NAME('bla-bla-bla') As ID From Dual;
```

возвратит null.

Создание параметров объекта IMan

```
declare
 -- Параметр объекта модели.
 mp SP.TMPAR;
 -- ID объекта модели
 MObjID number;
begin
 --Находим ID объекта по его полному имени
 MObjID := SP.MO.MOD_OBJ_ID_BY_FULL_NAME
      ('/Центр им. Д. Рогачёва/Модели/LIDAR_DATA/TestLidarData');
 -- Создаем параметр объекта с именем примечание
 mp := SP.TMPAR(MObjID,'Примечание');
 -- Задаём значение параметра объекта
 mp.VAL := S_('Первичные данные лазерного сканирования...');
 -- Сохраняем значение параметра объекта
 mp.save;
 commit;
end;
```

Проверка

Для проверки результатов достаточно выбрать все параметры объекта и распечатать их.

```
declare
 -- Параметр объекта модели.
 p SP.G.TMACRO_PARS;
 -- ID объекта модели
 MObjID number;
 idx VARCHAR2(128);
begin
 --Находим ID объекта по его полному имени
 MObjID := SP.MO.MOD_OBJ_ID_BY_FULL_NAME
      ('/Центр им. Д. Рогачёва/Модели/LIDAR_DATA/TestLidarData');
 -- Выбираем все параметры объекта в ассоциативный массив p.
 p := SP.MO.GET_MODEL_OBJECT(ObjectID => MObjID);
 idx := p.First;
 While Not idx Is Null Loop
   -- Печатаем значения всех элементов массива p.
   DBMS_OUTPUT.PUT_LINE(idx||' => '||p(idx).AsString);
   idx:=p.Next(idx);
 End Loop;
end;
```

Загрузка файла данных лазерного сканирования

Первичная загрузка данных лазерного сканирования является особым случаем создания параметра объекта Iman.

В настоящий момент используются следующие типы файлов данных лазерного сканирования:

1. las - LIDAR Data Exchange Format
2. pod - Bentley's Point Database files
3. rcp - Autodesk ReCap Project file

```
declare
 -- Параметр объекта модели.
 mp SP.TMPAR;
 -- ID объекта модели
 MObjID number;
 -- Тип загружаемого файла
 FType SP.TVALUE;
 --Указатель на BLOB в таблице SP.LOB_S
 vBLOB SP.TVALUE;
 --локатор бинарного большого объекта
 BiFile BLOB;
begin
 -- Передача бинарных данных из внешнего параметра в переменную BLOB
 BiFile:=:1;
 FType:=V_('FileType','las');
```

```
 --Находим ID объекта по его полному имени
 MObjID := SP.MO.MOD_OBJ_ID_BY_FULL_NAME
      ('/Центр им. Д. Рогачёва/Модели/LIDAR_DATA/TestLidarData');

 --Сохраняем файл в БД и возвращаем указатель на него
 vBLOB := SP.LOBS.LStore(BinFile =>BiFile,
                 tag => MObjID,
                 FileType => FType);
 -- Создаем параметр объекта с именем 'Файл'
 mp := SP.TMPAR(MObjID,'Файл');
 -- Задаём значение параметра объекта
 mp.VAL := vBLOB;
 -- Сохраняем значение параметра объекта
 mp.save;

 commit;
end;
```

Обновление файла данных лазерного сканирования

Обновление файла данных проходит по той же схеме, что первичная загрузка (см. раздел Загрузка файла данных лазерного сканирования). При этом в таблицу SP.LOB_S будет загружен новый экземпляр файла данных, сформирован новый GUID и новое значение переменной vBLOB, которое будет сохранено в том же параметре 'Файл'. В зависимости от настроек параметра 'Файл' старое значение будет сохранено в истории параметра, либо удалено. В последнем случае для окончательного удаления содержимого старого файла из таблицы SP.LOB_S требуется вызвать процедуру SP.LOBS.Delete_not_Used.

Выгрузка файла данных лазерного сканирования

Алгоритм выгрузки заключается в следующем:

- Находим карточку файла
- Находим параметр-ссылку на содержимое файла
- Извлекаем содержимое файла в OUTPUT-параметр

```
declare
 -- Параметр объекта модели (карточки файла).
 mp SP.TMPAR;
 -- ID объекта модели (карточки файла)
 MObjID number;
begin

 --Находим ID карточки файла по её полному имени
 MObjID := SP.MO.MOD_OBJ_ID_BY_FULL_NAME
      ('/Центр им. Д. Рогачёва/Модели/LIDAR_DATA/TestLidarData');

 -- Читаем параметр объекта с именем 'Файл'
 mp := SP.TMPAR(MObjID,'Файл');
 -- Передаём содержимое BLOB во внешний параметр
  :1:=SP.LOBS.getFile(FileRef => mp);

end;
```

Справочник

Функции

SP.LOBS.BL2Str

```
-- Проверка существования и получение TAG||GUID|TYPE файла.
FUNCTION BL2Str(V in SP.TVALUE) return VARCHAR2;
```

Параметр V должен быть типа 'BLOB', он может быть получен с помощью функций SP.LOBS.S2BLob или он может быть получен как значение параметра одной из карточек файла информационной модели.

Если значение V не задано (т.е. V.N и V.S пусты), то функция SP.LOBS.BL2Str возвращает пустое значение. Если значение V задано, но не указывает ни на одну из записей таблицы SP.LOB_S, то возникает исключение. В остальных случаях возвращается строка в одном из следующих форматов

Формат строки	**Условие**
GUID	Тег и тип файла не заданы
TAG||GUID	Тег задан, тип файла не задан
GUID|TYPE	Тег не задан, тип файла задан
TAG||GUID|TYPE	Тег и тип файла заданы

SP.LOBS.CL2Str

```
-- Проверка существования и получение TAG||GUID|TYPE файла.
```

```
FUNCTION CL2Str(V in SP.TVALUE) return VARCHAR2;
```

Параметр V должен быть типа 'CLOB', он может быть получен с помощью функции SP.LOBS.S2CLob или он может быть получен как значение параметра одной из карточек файла информационной модели. Описание возвращаемых значений см. в разделе SP.LOBS.BL2Str.

SP.LOBS.Delete_not_Used

```
-- Удаление файлов, на которые нет ссылок.
PROCEDURE Delete_not_Used;
```

SP.LOBS.getFile

```
-- Чтение файла по ссылке.
-- У пользователя должна быть роль на использования объекта,
-- которому принадлежит параметр.
-- Если задана дата, то ссылка на файл будет взята из истории параметра.
FUNCTION getFile(FileRef in SP.TMPAR, D in DATE default null) return BLOB;
```

SP.LOBS.getTxtFile

```
-- Чтение текстового файла по ссылке.
-- У пользователя должна быть роль на использования объекта,
-- которому принадлежит параметр
-- Если задана дата, то ссылка на файл будет взята из истории параметра.
FUNCTION getTxtFile(FileRef in SP.TMPAR, D in DATE default null) return CLOB;
```

SP.LOBS.LStore

```
-- Запись в базу нового файла, и получение ссылки на этот файл.
-- При этом поле X может содержать TAG файла,
-- а поле Y - идентифакатор его типа(TFileType).
FUNCTION LStore(BinFile in BLOB,
                tag in NUMBER default null,
                FileType in SP.TVALUE default null)
return SP.TVALUE;
```

SP.LOBS.LStore

```
-- Запись в базу нового текстового файла, и получение ссылки на этот файл.
-- При этом поле X может содержать TAG файла,
-- а поле Y - идентифакатор его типа(TFileType).
FUNCTION LStore(TxtFile in CLOB,
                tag in NUMBER default null,
```

```
                FileType in SP.TVALUE default null)
return SP.TVALUE;
```

SP.LOBS.S2BLob

```
-- Получение значения типа ссылки на файл по его GUID.
-- Строка может быть вида TAG||GUID|TYPE
-- При получении ссылки TAG и GUID игнорируются и всегда читаются из таблицы
-- Lobs
PROCEDURE S2BLob(S in VARCHAR2, V in out nocopy SP.TVALUE);
```

SP.LOBS.S2CLob

```
-- Получение значения типа ссылки на текстовый файл по его GUID.
-- Строка может быть вида TAG||GUID|TYPE.
-- При получении ссылки TAG и GUID игнорируются и могут отличаться от
-- значений таблицы SP.LOB_S
PROCEDURE S2CLob(S in VARCHAR2, V in out nocopy SP.TVALUE);
```

SP.LOBS.testLob

```
-- Проверка Clob и Blob
PROCEDURE testLob(V in SP.TVALUE);
```

Параметр V должен быть типа 'BLOB' или 'CLOB'. Процедура осуществляет проверку на валидность и при необходимости возбуждает исключения.

SP.MO.GET_MODEL_OBJECT

```
-- Функция извлекает объект со всеми его свойствами по его ID.
-- Функция не проверяет принадлежит ли объект текущей модели!
FUNCTION GET_MODEL_OBJECT(ObjectID in NUMBER) return SP.G.TMACRO_PARS;
```

SP.MO.GET_MODEL_OBJECT

```
-- Функция извлекает объект.
-- Функция не проверяет принадлежит ли объект текущей модели!
-- Если ObjectID - null, то ищем объект по свойствам объекта, находящимся в
-- составне массива PARS.
-- При этом параметры ID и OID имеют приоритет перед полным именем при
-- определении объекта, если они определены.
-- Если параметр TINY равен null, то используем свойство TINY из массива PARS.
PROCEDURE GET_MODEL_OBJECT(PARS in out nocopy  SP.G.TMACRO_PARS,
                           ObjectID in NUMBER,
                           TINY in BOOLEAN);
```

SP.MO.GET_MODEL_OBJECT_PARS

```
-- Функция предоставляет набор параметров объекта в виде курсора.
-- Names - список имён параметров, которые необходимы
-- или нулл, чтобы получить все
FUNCTION GET_MODEL_OBJECT_PARS(Object_ID in NUMBER,
                               Names in SP.TSTRINGS default null)
return TPars pipelined;
```

SP.MO.GET_V_MODEL_OBJECT_PARS

```
-- Функция предоставляет расширенный набор параметров объекта в виде курсора.
-- Может использоваться для организации представлений редактирования параметров
FUNCTION GET_V_MODEL_OBJECT_PARS(Object_ID in NUMBER)
return TObjectPars pipelined;
```

SP.MO.MOD_OBJ_ID_BY_FULL_NAME

```
-- Получение идентификатора объекта модели, по его полному имени.
-- Если объект - корень иерархии, то его идентификатор будет "1".
-- Если имя относительное, то при поиске объекта будет учтён текущий
-- опорный объект.
-- Если объект по имени не найден, то возвращается null.
-- Текущая модель берётся из глобального параметра.
FUNCTION MOD_OBJ_ID_BY_FULL_NAME(FullName in VARCHAR2)
return NUMBER;
```

Типы

SP.G.TMACRO_PARS

```
-- Параметры в пакетах макропроцедур.
TYPE TMACRO_PARS IS TABLE OF SP.TVALUE INDEX BY VARCHAR2(128);
```

SP.TMPAR

```
create or replace TYPE    TMPAR
/* Параметр объекта модели.*/
/* SP-TPAR.tps*/
AS OBJECT
(
/* Идентификатор объекта модели*/
MO_ID NUMBER,
/* Идентификатор параметра.
   Если идентификатор параметра объекта модели равен нулл,
   значит значение этого параметра равно значению по умолчанию.
   Процедура Save при изменении значения добавит параметр в таблицу.
   Если параметр уже в таблице, то он не будет удалён, даже если его значение
```

```
   после редактирования станет равно значению по умолчанию.*/
MP_ID NUMBER,
/* Идентификатор параметра в каталоге.
   Если идентификатор параметра объекта каталога равен нулл,
   значит этот параметр определён в сторонней модели.*/
CP_ID NUMBER,
/* Имя параметра.*/
NAME VARCHAR2(128),
/* Модификатор записи для параметра определённого в каталоге.*/
R_ONLY NUMBER(1),
/* Значение параметра.*/
VAL SP.TVALUE,
/* Дата создания или изменения параметра объекта. */
MDATE DATE,
/* Пользователь, создавший или изменивший параметр объекта. */
MUSER VARCHAR2(60),
/* При обновлении значения параметра будет автоматически проставлена текущая
дата в поле дата изменения параметра. Если мы хотим присвоить дату отличную от
текущей, то необходимо присвоить этому полю значение отличное от нулл. */
NEW_MDATE DATE,
/* При обновлении значения параметра будет автоматически заполнено поле,
содержащее имя пользователя, изменившего параметр. Если мы хотим присвоить
этому
полю значение отличное от имени текущего пользователя, то необходимо присвоить
этому полю значение отличное от нулл. */
NEW_MUSER VARCHAR2(60),
/* Идентификатор группы к которой принадлежит параметр. (Только для чтения.)*/
G_ID NUMBER,
/* Создание параметра модели и заполнение его значения по его идентификатору в
таблице параметров модели.*/
CONSTRUCTOR FUNCTION TMPAR(ModelParID IN NUMBER)
RETURN SELF AS RESULT,
/* Создание параметра модели и заполнение его значения по идентификатору
объекта модели и имени параметра.*/
CONSTRUCTOR FUNCTION TMPAR(ModelObjID IN NUMBER, Par IN VARCHAR2)
RETURN SELF AS RESULT,
/* Создание параметра модели и заполнение его значения по стороннему
идентификатору объекта модели, имени параметра и идентификатора модели.
Если идентификатор модели - NULL, то используем текущую модель.*/
CONSTRUCTOR FUNCTION TMPAR(ModelObjOID IN VARCHAR, Par IN VARCHAR2,
                           ModelID IN NUMBER DEFAULT NULL)
RETURN SELF AS RESULT,
/* Функция возвращает полное имя объекта модели.*/
MEMBER FUNCTION OBJECT_NAME RETURN VARCHAR2,
/* Функция возвращает имя группы к которой принадлежит параметр.*/
MEMBER FUNCTION GROUP_NAME RETURN VARCHAR2,
/* Сохранение значения параметра в таблицу параметров. Процедура заполняет
идентификатор параметра, если параметр имел значение по умолчанию.
Если не установлен параметр BlockHistory, то старое значение будет занесено
в историю.*/
MEMBER PROCEDURE Save(self IN OUT NOCOPY SP.TMPAR,
                      BlockHistory in BOOLEAN default false),
/* Сохранение значения входного параметра "V" в таблицу параметров в качестве
нового значения параметра, если значение "V" не совпадает с текущим значением
параметра self.Val. Процедура заполняет идентификатор параметра, если параметр
имел значение по умолчанию.*/
```

```
MEMBER PROCEDURE Save(self IN OUT NOCOPY SP.TMPAR, V in SP.TVALUE)
);
```

SP.TVALUE

```
create or replace TYPE    TValue
/* Универсальное значение для параметров оъектов каталога или модели,
а также глобальных параметров. Если глобальный параметр Check_ValEnabled не
установлен, то проверка значения не производится во всех членах типа! */
/* SP-TValue.tps*/
AS OBJECT
(
/* Тип параметра.*/
T  NUMBER,
/* Поле может содержать описание для параметра или самого значения.*/
COMMENTS VARCHAR2(4000),
/* Данное поле используется при организации диалога с пользователем в команде
Get_User_Input. Если R_ONLY = 0, то значение параметра можно читать и
записывать. Если R_ONLY = 1, то значение параметра можно только читать. Если
R_ONLY = -1, то значение параметра должно быть обязательно обновлено
пользователем. */
R_ONLY NUMBER(1),
/* Если тип имеет конечный набор значений, то поле содержит имя текущего
значения. Это же значение и есть представление данного значения в виде
строки.*/
E  VARCHAR2(128),
/* Часть значения.*/
N  NUMBER,
/* Часть значения.*/
D  DATE,
/* Часть значения.
Как правило не совпадает с представлением значения в виде строки.*/
S  VARCHAR2(4000),
/* Часть значения.*/
X  NUMBER,
/* Часть значения.*/
Y  NUMBER,
--
/* Конструктор для создания нулл значения.*/
CONSTRUCTOR FUNCTION TValue
RETURN SELF AS RESULT,
/* Конструктор проверяет существование типа значения.*/
CONSTRUCTOR FUNCTION TValue(ValueType IN VARCHAR2)
RETURN SELF AS RESULT,
/* Конструктор проверяет существование типа значения.*/
CONSTRUCTOR FUNCTION TValue(ValueType IN NUMBER)
RETURN SELF AS RESULT,
/* Конструктор проверяет значение и, если оно не подходит и установлен атрибут
"Safe" != 0, то конструктор установит первое значение из набора значений,
определённого для данного типа.*/
CONSTRUCTOR FUNCTION TValue(ValueType IN VARCHAR2,StrValue IN VARCHAR2,
                            Safe IN NUMBER DEFAULT 0)
RETURN SELF AS RESULT,
/* Конструктор проверяет значение и, если оно не подходит и установлен атрибут
```

```
"Safe" != 0,  то конструктор установит первое значение из набора значений,
определённого для данного типа.*/
CONSTRUCTOR FUNCTION TValue(ValueType IN NUMBER,StrValue IN VARCHAR2,
                            Safe IN NUMBER DEFAULT 0)
RETURN SELF AS RESULT,
/* Конструктор проверяет существование типа значения и самого значения.*/
CONSTRUCTOR FUNCTION TValue(ValueType IN VARCHAR2,NumValue IN NUMBER)
RETURN SELF AS RESULT,
/* Конструктор проверяет существование типа значения и самого значения.*/
CONSTRUCTOR FUNCTION TValue(ValueType IN NUMBER,NumValue IN NUMBER)
RETURN SELF AS RESULT,
/* Конструктор проверяет тип значения и, если тип имеет именованные значения,
то находит добавляет имя значения. Если параметр "DisN" равен нулл или "1",
то поле дата будет нулл вне зависимости от параметра "D".
Конструктор проверяет значение и, если оно не подходит и установлен атрибут
"Safe" != 0,  то конструктор установит первое значение из набора значений,
определённого для данного типа. Для ссылочного типа, если заполнено поле S, но
отсутствует ссылка на идентификатор объекта в модели, то конструктор пытается
его найти. Если ссылка недействительна, то будет возбуждена ошибка.*/
CONSTRUCTOR FUNCTION TValue(ValueType IN NUMBER,
                            N    IN NUMBER,
                            D    IN DATE DEFAULT NULL,
                            DisN IN NUMBER DEFAULT 1,
                            S    IN VARCHAR2,
                            X    IN NUMBER,
                            Y    IN NUMBER,
                            Safe IN NUMBER DEFAULT 0)
RETURN SELF AS RESULT,
/*Конструктор проверяет существование типа значения и самого значения.
Если параметр "DisN" равен нулл или "1", то поле дата будет нулл вне
зависимости от параметра "D".Для ссылочного типа, если заполнено поле S, но
отсутствует ссылка на идентификатор объекта в модели, то конструктор пытается
его найти. Если ссылка недействительна, то будет возбуждена ошибка.*/
CONSTRUCTOR FUNCTION TValue(ValueType IN NUMBER,
                            E    IN VARCHAR2,
                            N    IN NUMBER,
                            D    IN DATE DEFAULT NULL,
                            DisN IN NUMBER DEFAULT 1,
                            S    IN VARCHAR2,
                            X    IN NUMBER,
                            Y    IN NUMBER)
RETURN SELF AS RESULT,
--
/* Функция предоставляет для сравнения строку  состоящую из идентификатора
   типа и преобразованного к строке значения.
   Возможна ошибка. Если строки длинные и отличаются в последних байтах,
   то будут равны!*/
MAP MEMBER FUNCTION map_values(self IN OUT NOCOPY SP.TVALUE) RETURN VARCHAR2,
/* Функция возвращает имя типа значения.*/
MEMBER FUNCTION TypeName RETURN VARCHAR2,
/* Функция возвращает значение в виде строки.*/
MEMBER FUNCTION asString(self IN OUT NOCOPY SP.TVALUE) RETURN VARCHAR2,
/* Функция возвращает значение в виде логического типа, если это возможно,
иначе возникает прерывание.*/
MEMBER FUNCTION asBoolean RETURN BOOLEAN,
/* Функция возвращает значение в виде логического типа, если это возможно,
```

```
иначе возникает прерывание.*/
MEMBER FUNCTION B RETURN BOOLEAN,
/* Процедура заполняет все поля значения на основании его строкового
представления, если это возможно. Если значение не подходит и установлен флаг
"Safe" = true,  то процедура установит первое значение из набора значений,
определённого для данного типа. Если это невозможно или флаг "Safe" != true
то возникает прерывание.*/
MEMBER PROCEDURE Assign(self IN OUT NOCOPY SP.TVALUE, StrValue IN VARCHAR2,
                        Safe in BOOLEAN default false),
/* Процедура заполняет все поля значения на основании логического значения,
если это возможно, иначе возникает прерывание.*/
MEMBER PROCEDURE Assign(self IN OUT NOCOPY SP.TVALUE, BoolValue IN BOOLEAN),
/* Процедура заполняет все поля значения на основании числового значения,
если это возможно, иначе возникает прерывание.*/
MEMBER PROCEDURE Assign(self IN OUT NOCOPY SP.TVALUE, NumValue IN NUMBER),
/* Процедура копирует все поля значения.*/
MEMBER PROCEDURE Assign(self IN OUT NOCOPY SP.TVALUE, Val IN SP.TVALUE),
/* Процедура устанавливает поле R_ONLY = 1.*/
MEMBER PROCEDURE READ_ONLY(self IN OUT NOCOPY SP.TVALUE),
/* Процедура устанавливает поле R_ONLY = 0.*/
MEMBER PROCEDURE READ_WRITE(self IN OUT NOCOPY SP.TVALUE),
/* Процедура устанавливает поле R_ONLY = -1.*/
MEMBER PROCEDURE REQUIRED(self IN OUT NOCOPY SP.TVALUE)
);
```

Таблицы и представления

SP.LOB_S

Таблица непосредственно недоступна для пользователей. Взаимодействие с ней осуществляется посредством функций пакета SP.LOBS.

Наименование	**Тип**	**Допускает Null**	**Описание**
ID	NUMBER	No	Идентификатор записи
GUID	VARCHAR2(40)	Yes	Глобальный идентификатор файла
F_CLOB	CLOB	Yes	Текстовый файл
F_BLOB	BLOB	Yes	Бинарный файл
M_DATE	DATE	Yes	Дата создания или изменения записи.
M_USER	VARCHAR2(60)	Yes	Пользователь, создавший или изменивший запись.
TAG	NUMBER	Yes	Tag. Произвольное число.
F_TYPE	NUMBER	Yes	Тип файла. Тип определяется, ка идентификатор именованного значения TFileType

SP.V_GLOBAL_PAR_S

Наименование	Тип	Допускает Null	Описание
ID	NUMBER	No	Идентификатор параметра.
NAME	VARCHAR2(128)	No	Имя параметра. Имя параметра, является его идентификатором для клиента.
TYPE_ID	NUMBER(9)	No	Идентификатор типа параметра. 1..999 - встроенные типы. Такие типы может добавлять и редактировать только PROG.
TYPE_NAME	VARCHAR2(128)	Yes	Имя типа параметра.
VALUE_ENUM	NUMBER(1)	Yes	Признак именованного значения. 0 - значение не имеет имени, 1 - значение именовано.
SET_OF_VALUES	NUMBER(1)	Yes	Признак наличия у типа списка выбора для значения. 0 - значение не имеет списка выбора, 1 - значение имеет список выбора.
S_VALUE	VARCHAR2(4000)	Yes	Представление значение параметра в виде уникальной строки.
VALUE_COMMENTS	VARCHAR2(4000)	Yes	Комментарий к значению параметра.
REACTION	VARCHAR2(4000)	Yes	Блок PL/SQL, выполняемый после обновления параметра. При установке соединения выполняются все существующие блоки после заполнения рабочей таблицы параметров, причём порядок

Наименование	**Тип**	**Допускает Null**	**Описание**
			выполнения не определён и необходимо внутри блока, проводить проверку значений взаимоувязанных параметров и выполнять необходимы (повторные) дейстрия, например, как при установке территории сессии. Внутри блока доступен параметр "P" типа "SP.TGPAR", содержащий новое значение.
R_ONLY	VARCHAR2(30)	Yes	Если 1, то это параметр только для чтения из сессии пользователя из других сессий его можно менять. При создании типа TGPar данного параметра в сессии пользователя будет использовано значение из постояной таблицы SP.GLOBAL_PAR_S(с учётом возможного переопределения значения в значения в SP.USERS_GLOBALS), 2 - значения параметра неизменно для базы в целом. -1 нельзя менять блок реакции, но можно записывать значение параметра. Значения -1 и 2 доступны только для встроенных параметров.
R_ONLY_ID	NUMBER(3)	Yes	
COMMENTS	VARCHAR2(4000)	No	Описание параметра.
GROUP_ID	NUMBER	No	Группа параметра.
GROUP_NAME	VARCHAR2(128)	Yes	Имя группы, которой принадлежит данный параметр.

SP.V_GLOBALS

Наименование	Тип	Допускает Null	Описание
NAME	VARCHAR2(128)	No	Имя параметра.
TYPE_ID	NUMBER(9)	No	Идентификатор типа параметра. 1..999 - встроенные типы. Такие типы может добавлять и редактировать только PROG.
TYPE_NAME	VARCHAR2(128)	Yes	Имя типа параметра.
VALUE_ENUM	NUMBER(1)	Yes	Признак именованного значения. 0 - значение не имеет имени, 1 - значение именовано.
SET_OF_VALUES	NUMBER(1)	Yes	Признак наличия у типа списка выбора для значения. 0 - значение не имеет списка выбора, 1 - значение имеет список выбора.
S_VALUE	VARCHAR2(4000)	Yes	Представление значение параметра в виде уникальной строки
COMMENTS	VARCHAR2(4000)	Yes	Комментарий к глобальному параметру и его типу, а для именованных типов ещё и к значению параметра.
REACTION	VARCHAR2(4000)	Yes	Блок pl/sql, выполняемый после обновления параметра.
R_ONLY	VARCHAR2(4000)	Yes	Если >0 то это параметр только для чтения.
E	VARCHAR2(128)	Yes	Имя значения перечисляемого типа.

Наименование	Тип	Допускает Null	Описание
N	NUMBER	Yes	Значение.
D	DATE	Yes	Значение.
S	VARCHAR2(4000)	Yes	Значение.
X	NUMBER	Yes	Значение.
Y	NUMBER	Yes	Значение.

SP.V_MODEL_OBJECT_PARS

Наименование	Тип	Допускает Null	Описание
ID	NUMBER	Yes	Идентификатор параметра объекта модели.
OBJ_ID	NUMBER	Yes	Ссылка на объект каталога.
PARAM_NAME	VARCHAR2(128)	Yes	Имя параметра.
OBJ_PAR_ID	NUMBER	Yes	Идентификатор параметра объекта в каталоге.
MOD_OBJ_ID	NUMBER	Yes	Идентификатор объекта модели.

Наименование	**Тип**	**Допускает Null**	**Описание**
TYPE_ID	NUMBER	Yes	Тип параметра.
TYPE_NAME	VARCHAR2(4000)	Yes	Имя типа.
R_ONLY_ID	NUMBER	Yes	Если значение равно 1, то этот параметр только для чтения. Если значение равно -1, то этот параметр обязательно должен быть присвоен перед вызовом команды. Значение по умолчанию из каталога можно использовать только для справки. Если значение -2, то история изменения значений параметра не записывается.История так же не записывается для параметров только для чтения.
R_ONLY	VARCHAR2(4000)	Yes	Имя значения модификатора доступа R_ONLY_ID
VALUE_ENUM	NUMBER	Yes	Признак именованного значения. 0 - значение не имеет имени, 1 - значение именовано.
SET_OF_VALUES	NUMBER	Yes	Признак наличия у типа списка выбора для значения. 0 - значение не имеет списка выбора, 1 - значение имеет список выбора.
D_VAL	VARCHAR2(4000)	Yes	Поле содержит значение по умолчанию для данного параметра в виде строки.

Наименование	**Тип**	**Допускает Null**	**Описание**
VAL	VARCHAR2(4000)	Yes	Значение для данного параметра в виде строки.
E_VAL	VARCHAR2(128)	Yes	Значение перечисляемого типа.
N	NUMBER	Yes	Значение.
D	DATE	Yes	Значение.
S	VARCHAR2(4000)	Yes	Значение.
X	NUMBER	Yes	Значение.
Y	NUMBER	Yes	Значение.
ISREDEFINE	NUMBER	Yes	Если параметр имеет значение по умолчанию, то значение данного поля "0", если значение переопределено, то - "1".
M_DATE	DATE	Yes	Дата создания или изменения параметра объекта модели.
M_USER	VARCHAR2(60)	Yes	Пользователь создавший или изменивший параметр объект модели.

SP.V_MODEL_OBJECTS

Наименование	Тип	Допускает Null	Описание	Допускает изменения
ID	NUMBER	No	Идентификатор объекта. У объекта есть предопределённый параметр "ID".	YES
OID	VARCHAR2(40)	Yes	Идентификатор объекта во внешней модели. У объекта есть предопределённый параметр "OID". Для локальных моделей идентификатор присваиается при добавлении объекта средствами базы.	YES
PARENT_MOD_OBJ_ID	NUMBER	Yes	Ссылка на родительский объект. У объекта есть предопределённый параметр "PARENT"	YES
POID	VARCHAR2(40)	Yes	Сторонний уникальный идентификаторродителя.	YES
OBJ_ID	NUMBER	No	Ссылка на объект каталога.	YES
MODEL_ID	NUMBER	No	Идентификатор модели.	YES
MODEL_NAME	VARCHAR2(4000)	No	Имя модели.	YES
MOD_OBJ_NAME	VARCHAR2(128)	No	Имя объекта модели. Может быть присвоено генератором имён.	YES

Наименование	**Тип**	**Допускает Null**	**Описание**	**Допускает изменения**
			При добавлении, изменении или удалении объекта можно использовать полное имя объекта, включающее в себя путь. Если используется полное имя, то путь игнорируется. Можно также использовать относительное имя. В этом случае оно соединяется с полем "PATH".	
PATH	VARCHAR2(4000)	Yes	Полный путь к объекту модели.	NO
FULL_NAME	VARCHAR2(4000)	Yes	Полное имя объекта модели.	NO
OBJ_LEVEL	VARCHAR2(40)	Yes	Уровень вложенности объекта.	NO
KIND	VARCHAR2(4000)	Yes	Описание вида объекта.	NO
KIND_ID	NUMBER(1)	No	Вид объекта. (0 - одиночный объект каталога, 1 - композитный объект каталога, 2 - макропроцедура, 3 - одиночная операция(выполняемая сервером модели))	YES
CATALOG_NAME	VARCHAR2(128)	No	Имя объекта каталога.	YES
CATALOG_GROUP_NAME	VARCHAR2(128)	No	Имя группы объекта каталога. Уникальными в каталоге,	YES

Наименование	**Тип**	**Допускает Null**	**Описание**	**Допускает изменения**
			является пара "имя группы"+"имя объекта".	
CATALOG_COMMENTS	VARCHAR2(4000)	No	Описание объекта каталога.	YES
USING_ROLE_NAME	VARCHAR2(30)	Yes	Роль, которую должен иметь пользователь, чтобы изучать объект и его свойства. Пользователю, имеющему SP_ADMIN_ROLE доступен любой объект. Если поле нулл, то объект публичен. Любой пользователь может создать новый объект, но он может назначить ему роли, только которые имеет сам. Роль пользователя есть зарезервированный параметр объекта "USING_ROLE".	YES
USING_ROLE_ID	NUMBER	Yes	Идентификатор роли использования объекта модели.	YES
EDIT_ROLE_NAME	VARCHAR2(30)	Yes	Роль, которую должен иметь пользователь, чтобы изменять объект, а также добавлять, удалять или изменять его параметры. Пользователь, имеющий SP_ADMIN_ROLE, может изменять любой объект. Если поле нулл, то, в отличие от каталога, любой пользователь может изменять объект. Роль пользователя есть зарезервированный параметр объекта "EDIT_ROLE".	YES

Наименование	**Тип**	**Допускает Null**	**Описание**	**Допускает изменения**
EDIT_ROLE_ID	NUMBER	Yes	Идентификатор роли изменения объекта модели.	YES
M_DATE	DATE	No	Дата создания или изменения объекта модели.	YES
M_USER	VARCHAR2(60)	No	Пользователь создавший или изменивший объект модели.	YES

Источники

ORA1 Oracle Database SecureFiles and Large Objects Developer's Guide, 21c, 2020