

Кодирование типов данных 1С:Предприятие 8 в protobuf

0. Структура пакетов.

Для каждой информационной базы 1С должен создаваться соответствующий набор пакетов protobuf. Это необходимо для корректного отражения и кодирования системы типов информационной базы 1С в protobuf.

В свою очередь набор типов данных, участвующих в интеграции, определяется составом плана обмена 1С. Планов обмена в информационной базе 1С может быть несколько. Другими словами для каждого плана обмена необходимо формировать свой набор пакетов protobuf.

Кроме этого следует учитывать, что разные типы объектов 1С могут иметь одни и те же наименования. Например, справочник и регистр сведений могут иметь одно и то же наименование в рамках одной и той же информационной базы 1С.

Более того, табличные части различных объектов 1С так же могут иметь одинаковые наименования (вложенные типы).

Шаблон идентификации отдельно взятого типа данных 1С:

<Конфигурация>.<ПланОбмена>.**v**<ВерсияСхемы>.<ВидОбъекта>.<ИмяТипа>

Пример: ERP.ПланОбменаKafka.v1.Справочник.Номенклатура

В целях избежания дублирования наименований табличных частей целесообразно использовать вложенные типы данных protobuf. Например, для табличной части "Товары" документа "ЗаказКлиента" наименование типа protobuf будет выглядеть следующим образом:

ERP.ПланОбменаKafka.v1.Документ.ЗаказКлиента.Товары

Учитывая тот факт, что объекты 1С могут ссылаться друг на друга, пакеты protobuf информационной базы должны так же ссылаться друг на друга, используя **import** соответствующих файлов proto.

Структура каталогов файлов для описания схемы protobuf (пример):

```
- ERP
---- ПланОбменаKafka
----- v1
----- Документ.proto
----- Справочник.proto
----- РегистрСведений.proto
----- РегистрНакопления.proto
```

1. Простые типы данных.

1C	proto3
Неопределенно	google.protobuf.NullValue
Булево	bool
Число	double
Строка	string
Дата	google.protobuf.Timestamp
Уникальный Идентификатор (uuid)	string
Бинарные данные	bytes

2. Общие типы данных.

<pre> syntax = "proto3"; package 1c.protobuf.common; </pre>	
1C	proto3
<p>Вид Движения {Приход, Расход}</p> <p>Предопределённое платформой 1C перечисление, используемое регистрами накопления остатков.</p>	<pre> enum RecordType { Receipt = 0; Expense = 1; } </pre>
<p>Ссылка, структура {int:uuid}</p> <p>Ссылка может быть "пустой", то есть иметь значение нулевого UUID: "00000000-0000-0000-0000-000000000000"</p> <p>Свойство type содержит полное наименование типа данных 1C, например, "Справочник.Номенклатура".</p>	<pre> message Entity { string type = 1; string uuid = 2; } </pre> <p>Свойство uuid содержит значение внутреннего идентификатора объекта в базе данных 1C.</p>
<p>Составной тип</p> <p>Составной тип данных 1C полностью соответствует семантике oneof.</p> <p>Может не иметь установленных значений, то есть иметь значение Неопределенно в 1C.</p>	<pre> message Union { oneof value { NullValue undefined = 1; bool boolean = 2; double numeric = 3; string string = 4; Entity entity = 5; Timestamp datetime = 6; } } </pre>
<p>Удаление Объекта</p> <p>Информация об удалении объекта из информационной базы 1C.</p>	<pre> message ObjectDeletion { Entity entity = 1; } </pre>

2. Сложные типы данных.

2.1. Ссылочные типы данных.

Документы и справочники 1С кодируются в protobuf обычным способом как **message**. При этом для описания их свойств могут быть использованы только все выше описанные простые типы данных.

```
syntax = "proto3";

package ERP.ПланОбменаKafka.v1.Справочник;

import "1c/protobuf/common.proto";

message ДоговорыКонтрагентов
{
    string Ссылка          = 1;
    string Код             = 2;
    string Наименование   = 3;
    1c.protobuf.Entity Владелец = 4;
}
```

2.2. Табличные части (вложенные типы).

Табличные части могут иметь только ссылочные типы данных 1С. Они являются неотъемлемой частью своих объектов, образуя таким образом агрегаты. Кодируются как вложенные типы protobuf.

```
message ДоговорыКонтрагентов
{
    message КонтактнаяИнформация
    {
        string НомерТелефона    = 1;
        string ЭлектроннаяПочта = 2;
    }
    repeated КонтактнаяИнформация Контакты = 1;
}
```

2.3. Наборы записей регистров.

Регистры сведений и накопления в 1С всегда изменяются при помощи наборов записей. Набор записей кодируется как объект, имеющий два свойства: **delete** (отбор) и **insert** (записи набора).

Свойство **delete** содержит ключ записи регистра, по которому выполняется операция удаления. Значение ключа может быть не указано, что означает удаление всех записей таблицы без отбора.

Свойство **insert** содержит набор записей регистра, которые необходимо вставить в таблицу после выполнения удаления по ключу. Свойство **insert** может быть пустым, то есть не иметь записей.

Пример описания схемы для регистра сведений "КурсыВалют":

```
syntax = "proto3";

package ERP.ПланОбменаKafka.v1.РегистрСведений;

import "lc/protobuf/common.proto";
import "google/protobuf/struct.proto";
import "google/protobuf/timestamp.proto";

message КурсыВалют
{
    message Ключ
    {
        google.protobuf.NullValue undefined = 1;
        google.protobuf.Timestamp Период = 2;
        lc.protobuf.Entity Валюта = 3;
    }
    message Запись
    {
        КурсыВалют.Ключ Ключ = 1;
        double Курс = 2;
    }
    repeated КурсыВалют.Ключ delete = 1;
    repeated КурсыВалют.Запись insert = 2;
}
```

4. Обязательные заголовки сообщений.

Один топик Kafka может хранить сообщения разных типов для одного и того же объекта метаданных lC. Например, для справочника "Номенклатура" может существовать запись, как для обычного объекта, так и для типа "УдалениеОбъекта".

Таким образом, целесообразно дополнять сообщения Kafka специальным заголовком, например, **message-type** с указанием его типа.

Кроме этого, необходимо дополнять сообщения значением идентификатора (версии) соответствующей схемы protobuf, например, **schema-version**.