```
Процедура ЛогНачало(Лог,Наименование) Экспорт

	Действие    = Лог.Действия.Добавить();
	Действие.Наименование  = Наименование;
	Действие.Начало  = ТекущаяУниверсальнаяДатаВМиллисекундах();

КонецПроцедуры

Процедура ЛогКонец(Лог,КоличествоДействий) Экспорт

	Действие    = Лог.Действия[Лог.Действия.Количество()-1];
	Действие.Конец   = ТекущаяУниверсальнаяДатаВМиллисекундах();
	Действие.Длительность  = Действие.Конец-Действие.Начало;
	Действие.КоличествоДействий = КоличествоДействий;
	Лог.Записать();

КонецПроцедуры

Процедура ДополнитьЛог(Лог,Таймер,Сообщение)

	Лог   = Лог+" "+( (ТекущаяУниверсальнаяДатаВМиллисекундах()-
Таймер)/1000 )+" сек. "+Символы.ПС+ТекущаяДата()+" "+Сообщение;
	Таймер=ТекущаяУниверсальнаяДатаВМиллисекундах();

КонецПроцедуры

//Функция СтрокаПодключенияMySQL()
//
//    СтрокаПодключения      = "";
//
//    ВыполнятьСтандартно    = Истина;
//    ИмяФункцииДляИсполнения=("А_Питон.ИзЗапросаВПитон_ML");
//    Ответ="";
//
	Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));
//
//    Если ВыполнятьСтандартно Тогда
//
//          СтрокаПодключения = "Provider=SQLOLEDB;
//          |Initial Catalog=1c_python;
//          |Integrated Security=SSPI;
//          |User ID=ICECORP\1csystem;
//          |Password=;
//          |Data Source=10.2.4.124";
//
//    КонецЕсли;
//
//
	Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));
//
//    Возврат СтрокаПодключения;
//
//КонецФункции

//Процедура ИзЗапросаВПитон_ML(ИдентификаторДанных,ИдентификаторЛога)
```

```
//
//    ВыполнятьСтандартно    = Истина;
//    ИмяФункцииДляИсполнения=("А_Питон.ИзЗапросаВПитон_ML");
//    Ответ="";
//
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));
//
//    Если ВыполнятьСтандартно Тогда
//
//          //====
//          // Запрос: Загрузка данных
//          //====
//          Лог   =
Справочники.А_ЛогПроизводительности.НайтиПоКоду(ИдентификаторЛога).Получи
тьОбъект();

//          Элемент     =
Справочники.ML_Data.НайтиПоКоду(ИдентификаторДанных);

//          ЛогНачало(Лог,"Сервер: Запрос к 1С");
//          Запрос      = Новый Запрос;
//          Запрос.Текст     = Элемент.ТекстЗапроса;
//          Для Каждого ТекПараметр Из Элемент.ПараметрыЗапроса Цикл
//
      Запрос.УстановитьПараметр(ТекПараметр.ИмяПараметра,ТекПараметр.Знач
ениеПараметра);
//          КонецЦикла;
//          Результат  = Запрос.Выполнить();
//          ЛогКонец(Лог,1);

//          ЛогНачало(Лог,"Сервер: Формирование запроса на передачу в
Python");
//          Выборка     = Результат.Выбрать();
//
//          //Количество записей
//          ЭлементОбъект    = Элемент.ПолучитьОбъект();
//          ЭлементОбъект.КоличествоЗаписейРезультатаЗапроса   =
Выборка.Количество();
//          ЭлементОбъект.Записать();
//          Элемент     = ЭлементОбъект.Ссылка;
//
//          //Данные
//          ид_запроса  = Элемент.ИдентификаторЗапроса;
//          mysql_query= "";
//          ПервыйЗапрос     = Истина;
//          КоличествоЗапросов     =
Выборка.Количество()/?(Элемент.КатегориальныеПризнаки.Количество()>0,Элем
ент.КатегориальныеПризнаки.Количество(),1);
//          last_row    = 0;
//          ЛогКонец(Лог,1);
//          ЛогНачало(Лог,"Сервер: Отправка "+КоличествоЗапросов+"
запросов к Питону");
//          row=0;

//          //mysql fix1++
//          Connection = Новый COMОбъект("ADODB.Connection");
//          Connection.ConnectionString = СтрокаПодключенияMySQL();
//          Connection.Open();
```

```
//          //mysql fix1--

//          Пока Выборка.Следующий() Цикл
//
//                ИндексПоля = 0;
//
//                Для Каждого ТекПоле Из Результат.Колонки Цикл
//
//                      Если НЕ last_row=row Тогда
//                            //mysql fix2++
//                            Попытка
//                                  SqlQueryResult =
Connection.Execute("insert into
regression(request,row,field,value)"+mysql_query);
//                            Исключение
//
      А_Серверные.ЗаписатьСобытие(Элемент.Ссылка,""+row+":
"+ОписаниеОшибки(),,,Истина);
//                            КонецПопытки;
//                            //mysql fix2--
//                            last_row=row;
//                            mysql_query= "";
//                            ПервыйЗапрос     = Истина;
//
//                      КонецЕсли;
//
//                      Если
Найти(Строка(ТипЗнч(Выборка[ТекПоле.Имя])),"Число")>0 Тогда
//                            value = Формат(Выборка[ТекПоле.Имя],"ЧРД=.;
ЧН=; ЧГ=");
//                      Иначе
//                            value = Выборка[ТекПоле.Имя];
//                      КонецЕсли;
//
//                      mysql_query= mysql_query    + ?(ПервыйЗапрос," ","
union all ") + "select '"+ид_запроса+"',"+Формат(row,"ЧН=;
ЧГ=")+","+Формат(ИндексПоля,"ЧН=; ЧГ=")+",'"+value+"'";
//                      ПервыйЗапрос     = Ложь;
//
//                      ИндексПоля  = ИндексПоля+1;
//
//                КонецЦикла;
//
//                row=row+1;
//
//          КонецЦикла;

//          Если НЕ last_row=row Тогда
//
//                //mysql fix3++
//                Попытка
//                      SqlQueryResult = Connection.Execute("insert into
regression(request,row,field,value)"+mysql_query);
//                Исключение
//
      А_Серверные.ЗаписатьСобытие(Элемент.Ссылка,""+row+":
"+ОписаниеОшибки(),,,Истина);
//                КонецПопытки;
//                //mysql fix3--
//
```

```
//          КонецЕсли;

//          ЛогКонец(Лог,row);
//
//    КонецЕсли;
//
//
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));

//КонецПроцедуры

//Функция Обучение_ML(ИдентификаторДанных,ИдентификаторЛога) Экспорт
//
//    ВыполнятьСтандартно    = Истина;
//    ИмяФункцииДляИсполнения=("А_Питон.Обучение_ML");
//    Ответ="";
//
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));
//
//    Если ВыполнятьСтандартно Тогда
//
//          ИзЗапросаВПитон_ML(ИдентификаторДанных,ИдентификаторЛога);
//
//          Лог   =
Справочники.А_ЛогПроизводительности.НайтиПоКоду(ИдентификаторЛога).Получи
тьОбъект();
//          Элемент     =
Справочники.ML_Data.НайтиПоКоду(ИдентификаторДанных).ПолучитьОбъект();
//
//          ид_запроса  = Элемент.ИдентификаторЗапроса;
//
//          КатегориальныеПризнаки = "";
//          Для Каждого ТекСтрока Из Элемент.КатегориальныеПризнаки Цикл
//                Если ТекСтрока.Категориальный Тогда
//                      Если СтрДлина(КатегориальныеПризнаки)>0 Тогда
//
      КатегориальныеПризнаки=КатегориальныеПризнаки+",";
//                      КонецЕсли;
//
      КатегориальныеПризнаки=КатегориальныеПризнаки+ТекСтрока.Признак;
//                КонецЕсли;
//          КонецЦикла;
//
//          ЛогНачало(Лог,"Сервер: Обучение");
//          хттпЗапрос  = Элемент.АдресСкриптаОбучения
//                +"?model="+Элемент.Модель
//                +"&request="+ид_запроса
//                +"&iter_count="+Элемент.КоличествоИтераций
//
      +"&learning_rate="+Формат(Элемент.КоэффициентОбучения,"ЧРД=.; ЧН=;
ЧГ=")
//                +"&depth="+Элемент.ГлубинаОбуения
//                +"&data_file="+Элемент.ФайлДанных
//                +"&cat_features="+КатегориальныеПризнаки;
```

```
//
      А_Серверные.ВыполнитьHTTPЗапросПолучитьОтвет(хттпЗапрос,Ответ,Элеме
нт.ТаймаутЗапроса,Ложь);
//
//          Элемент.ЛогОбучения    = Ответ;
//
//          Попытка
//
//                ПоследняяСтрока  =
СтрПолучитьСтроку(Ответ,СтрЧислоСтрок(Ответ));
//
//                Элемент.КоэффициентДетерминации   =
Число(ПоследняяСтрока);
//
//          Исключение
//
      А_Серверные.ЗаписатьСобытие(ПараметрыСеанса.ТекущийПользователь,"По
следняя строка (КоэффициентДетерминации) не является числом:
"+ПоследняяСтрока,,,Истина);
//          КонецПопытки;
//
//          Элемент.Записать();
//
//          ЛогКонец(Лог,1);
//
//    КонецЕсли;
//
//
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));
//    Возврат(Ответ);
//
//КонецФункции

Процедура
ВыполнитьЗапросНаСервере_ML(ИдентификаторДанных,ИдентификаторЛога,Заполня
тьКатегории) Экспорт

      Лог   =
Справочники.А_ЛогПроизводительности.НайтиПоКоду(ИдентификаторЛога).Получи
тьОбъект();
      ЛогНачало(Лог,"Сервер: Запрос к 1С");
      Элемент     =
Справочники.ML_Data.НайтиПоКоду(ИдентификаторДанных).ПолучитьОбъект();

      //Запрос
      Запрос      = Новый Запрос;
      Запрос.Текст      = Элемент.ТекстЗапроса;
      Для Каждого ТекПараметр Из Элемент.ПараметрыЗапроса Цикл
            Если ТекПараметр.ТекущаяДата Тогда
            Попытка

      Запрос.УстановитьПараметр(ТекПараметр.ИмяПараметра,ТекущаяДата());
            Исключение
                  А_Серверные.ЗаписатьСобытие(Элемент, "Не удалось
установить параметр элемента ML Data: "+ОписаниеОшибки());

      Запрос.УстановитьПараметр(ТекПараметр.ИмяПараметра,ТекПараметр.Знач
ениеПараметра);
```

```
                  КонецПопытки;
            Иначе

      Запрос.УстановитьПараметр(ТекПараметр.ИмяПараметра,ТекПараметр.Знач
ениеПараметра);
            КонецЕсли;
      КонецЦикла;
      Результат  = Запрос.Выполнить();
      ЛогКонец(Лог,1);

      ЛогНачало(Лог,"Сервер: Выгрузка в ТЧ");

      Элемент.Data.Очистить();

      Выборка    = Результат.Выбрать();

      //Категории
      Если ЗаполнятьКатегории Тогда
            Элемент.КатегориальныеПризнаки.Очистить();
            Индекс     = 0;
            Для Каждого ТекПоле Из Результат.Колонки Цикл
                  НовыйПризнак                        =
Элемент.КатегориальныеПризнаки.Добавить();
                  НовыйПризнак.Признак        =
"field_"+Формат(Индекс,"ЧН=; ЧГ=");
                  НовыйПризнак.Псевдоним      = ТекПоле.Имя;
                  НовыйПризнак.ОписаниеТипа   =
Строка(ТекПоле.ТипЗначения);
                  Индекс     = Индекс+1;
            КонецЦикла;
      КонецЕсли;
      //Для Каждого ТекСтрока Из Data Цикл
      //    Если ТекСтрока.row>0 Тогда
      //          Прервать;
      //    КонецЕсли;
      //    НовыйПризнак     = КатегориальныеПризнаки.Добавить();
      //    НовыйПризнак.Признак   = "field_"+ТекСтрока.field;
      //КонецЦикла;

      //Данные
      row=0;
      Пока Выборка.Следующий() Цикл
            ИндексПоля = 0;
            Для Каждого ТекПоле Из Результат.Колонки Цикл
                  НовСтрока  = Элемент.Data.Добавить();
                  НовСтрока.row    = row;
                  НовСтрока.field  = ИндексПоля;

                  Если Найти(Строка(ТекПоле.ТипЗначения),"Число")>0 Тогда
                        НовСтрока.value  =
Формат(Выборка[ТекПоле.Имя],"ЧН=; ЧГ=");
                  Иначе
                        НовСтрока.value  = Выборка[ТекПоле.Имя];
                  КонецЕсли;
                  ИндексПоля = ИндексПоля+1;
            КонецЦикла;
            row=row+1;
      КонецЦикла;

      Элемент.Лог= Лог.Ссылка;
```

```
	Элемент.Записать();
	ЛогКонец(Лог,Элемент.Data.Количество());

КонецПроцедуры

//Функция Прогноз_ML(ИдентификаторДанных,ИдентификаторЛога) Экспорт
//
//	ВыполнятьСтандартно	= Истина;
//	ИмяФункцииДляИсполнения=("А_Питон.Прогноз_ML");
//	Ответ="";
//
	Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));
//
//	Если ВыполнятьСтандартно Тогда
//
//		ИзЗапросаВПитон_ML(ИдентификаторДанных,ИдентификаторЛога);
//
//		Лог	=
Справочники.А_ЛогПроизводительности.НайтиПоКоду(ИдентификаторЛога).Получи
тьОбъект();
//		Элемент	=
Справочники.ML_Data.НайтиПоКоду(ИдентификаторДанных).ПолучитьОбъект();
//
//		ЛогНачало(Лог,"Сервер: Прогноз "+Элемент.Модель);
//		Ответ="";
//		хттпЗапрос = Элемент.АдресСкриптаПрогноза
//			+"?model="+Элемент.Модель
//			+"&request="+Элемент.ИдентификаторЗапроса;
//
	А_Серверные.ВыполнитьHTTPЗапросПолучитьОтвет(хттпЗапрос,Ответ,Элеме
нт.ТаймаутЗапроса,Ложь);
//		ЛогКонец(Лог,1);
//		//====
//		// Чтение прогнозов питона
//		//====
//		//ПараметрыСоединения = Новый
ПараметрыСоединенияВнешнегоИсточникаДанных;
//		//ПараметрыСоединения.СтрокаСоединения = "DRIVER={SQL
Server};SERVER=10.2.4.124;DATABASE=1c_python;";//UID=ICECORP\1csystem;PWD
=;";
//
	//ВнешниеИсточникиДанных.MachineLearning.УстановитьОбщиеПараметрыСо
единения(ПараметрыСоединения);
//
	//ВнешниеИсточникиДанных.MachineLearning.УстановитьСоединение();

//
//		//ЛогНачало(Лог,"Сервер: Запись в РС");
//		//
//		////Определим поле результата
//		//Запрос	= Новый Запрос;
//		//Запрос.Текст	= "ВЫБРАТЬ
//		//			|	МАКСИМУМ(regression.field) КАК field
//		//			|ИЗ
//		//			|
	ВнешнийИсточникДанных.MachineLearning.Таблица.regression КАК
regression
//		//			|ГДЕ
```

```
//          //                  |   regression.request =
&ИдентификаторЗапроса";
//
      //Запрос.УстановитьПараметр("ИдентификаторЗапроса",Элемент.Идентифи
каторЗапроса);
//          //ПолеПрогноза   = Запрос.Выполнить().Выгрузить()[0].field;
//          //
//          //Запрос    = Новый Запрос;
//          //Запрос.Текст   = "ВЫБРАТЬ
//          //                  |   regression.row,
//          //                  |   regression.value
//          //                  |ИЗ
//          //                  |
      ВнешнийИсточникДанных.MachineLearning.Таблица.regression КАК
regression
//          //                  |ГДЕ
//          //                  |   regression.request =
&ИдентификаторЗапроса
//          //                  |   И regression.field = &ПолеПрогноза";
//
      //Запрос.УстановитьПараметр("ИдентификаторЗапроса",Элемент.Идентифи
каторЗапроса);
//
      //Запрос.УстановитьПараметр("ПолеПрогноза",Элемент.ПолеПрогноза);
//          ////Элемент.Data.Загрузить(Запрос.Выполнить().Выгрузить());
//          //
//          //Выборка   = Запрос.Выполнить().Выбрать();
//          ////Запись в РС++
//          //Пока Выборка.Следующий() Цикл
//          //
//          //    Набор = РегистрыСведений.CatBoost.СоздатьНаборЗаписей();
//          //
//          //    Набор.Отбор.id.Установить(Элемент.ИдентификаторЗапроса);
//          //    Набор.Отбор.Row.Установить(Выборка.row);
//          //    //Набор.Отбор.field.Установить(Выборка.field);
//          //    Набор.Отбор.field.Установить(Элемент.ПолеПрогноза);
//          //
//          //    НоваяЗапись= Набор.Добавить();
//          //
//          //    НоваяЗапись.id          = Элемент.ИдентификаторЗапроса;
//          //    НоваяЗапись.Row         = Выборка.row;
//          //    //НоваяЗапись.field     = Выборка.field;
//          //    НоваяЗапись.field       = Элемент.ПолеПрогноза;
//          //
//          //    НоваяЗапись.value       = Выборка.value;
//          //
//          //    Набор.Записать();
//          //
//          //КонецЦикла;
//          ////Запись в РС--
//          //ЛогКонец(Лог,Выборка.Количество());
//
//          //ЛогНачало(Лог,"Сервер: Запись Элемента");
//          //Элемент.Записать();
//          //ЛогКонец(Лог,1);
//          //
//          ////====
//          //// Удаление данных из MySQL
//          ////====
//          //ЛогНачало(Лог,"Сервер: Удаление данных из SQL ");
```

```
//          //Запрос    = "delete from regression where
request='"+Элемент.ИдентификаторЗапроса+"';";
//          //
//          //Connection = Новый COMОбъект("ADODB.Connection");
//          //Connection.ConnectionString = СтрокаПодключенияMySQL();
//          //Connection.Open();
//          //SqlQueryResult = Connection.Execute(Запрос);
//          //
//          //ЛогКонец(Лог,Элемент.Data.Количество());
//
//    КонецЕсли;
//
//
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));
//
//КонецФункции

Процедура ИнициализацияВнешнегоИсточника() Экспорт

      ПараметрыСоединения = Новый
ПараметрыСоединенияВнешнегоИсточникаДанных;
      ПараметрыСоединения.СтрокаСоединения    = "DRIVER={SQL
Server};SERVER=10.2.4.124;DATABASE=1c_python;";//UID=ICECORP\1csystem;PWD
=;";
      ВнешниеИсточникиДанных.MachineLearning.УстановитьОбщиеПараметрыСоед
инения(ПараметрыСоединения);
      ВнешниеИсточникиДанных.MachineLearning.УстановитьСоединение();


КонецПроцедуры

//Процедура УдалитьДанныеSQLПоЗапросу(ИдентификаторЗапроса) Экспорт
//
//    Запрос     = "delete from regression where
request='"+ИдентификаторЗапроса+"';";
//    Connection = Новый COMОбъект("ADODB.Connection");
//    Connection.ConnectionString = СтрокаПодключенияMySQL();
//    Connection.Open();
//    SqlQueryResult = Connection.Execute(Запрос);
//
//КонецПроцедуры

//Процедура УдалитьВсеДанныеSQL() Экспорт
//
//    //====
//    // Удаление данных из MySQL
//    //====
//    Запрос     = "delete from regression;";
//    Connection = Новый COMОбъект("ADODB.Connection");
//    Connection.ConnectionString = СтрокаПодключенияMySQL();
//    Connection.Open();
//    SqlQueryResult = Connection.Execute(Запрос);
//
//КонецПроцедуры

//Процедура ВычислитьВажность(ЭлементСсылка) Экспорт
//
//    Попытка
```

```
//
//          ВыполнятьСтандартно    = Истина;
//          ИмяФункцииДляИсполнения=("А_Питон.ВычислитьВажность");
//          Ответ="";
//
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));
//
//          Если ВыполнятьСтандартно Тогда
//
//                Ответ="";
//
      А_Серверные.ВыполнитьHTTPЗапросПолучитьОтвет("http://10.2.4.87:8000
/feature_importance?model="+ЭлементСсылка.Модель,Ответ,ЭлементСсылка.Тайм
аутЗапроса,Ложь);
//                Разделитель=" ";
//                СтрокаПреобразованнаяВМногострочныйТекст     =
СтрЗаменить(Ответ,Разделитель,Символы.ПС);
//                ТекстовыйДокументИзСтроки                    = Новый
ТекстовыйДокумент;
//
      ТекстовыйДокументИзСтроки.УстановитьТекст(СтрокаПреобразованнаяВМно
гострочныйТекст);
//                ЭлементОбъект    = ЭлементСсылка.ПолучитьОбъект();
//                Для СчетчикСтрок = 1 по
ТекстовыйДокументИзСтроки.КоличествоСтрок() цикл
//                      Запись           =
СокрЛП(ТекстовыйДокументИзСтроки.ПолучитьСтроку(СчетчикСтрок));
//                      Поле       = Лев(Запись,Найти(Запись,",")-1);
//                      Признак    =
ЭлементОбъект.КатегориальныеПризнаки.Найти(Поле,"Признак");
//                      Если НЕ Признак  = Неопределено Тогда
//                            Признак.Важность =
Число(Прав(Запись,Стрдлина(Запись)-СтрДлина(Поле)-1));
//                      КонецЕсли;
//                КонецЦикла;
//                ЭлементОбъект.Записать();
//
//          КонецЕсли;
//
//    Исключение
//          А_Серверные.ЗаписатьСобытие(ЭлементСсылка,"Вычисление
важности: ",,,Истина);
//    КонецПопытки;
//
//КонецПроцедуры

Процедура ИсключитьНули(ЭлементСсылка) Экспорт

      Попытка

            ПозицияГраницыЗапросов = 0;
            ЗапросЛеваяЧасть = "";
            ЗапросПраваяЧасть      = ЭлементСсылка.ТекстЗапроса;
            Пока Найти(ЗапросПраваяЧасть,";")>0 Цикл
                  Граница     = Найти(ЗапросПраваяЧасть,";");
                  ЗапросЛеваяЧасть =
ЗапросЛеваяЧасть+Лев(ЗапросПраваяЧасть,Граница);
```

```
                ЗапросПраваяЧасть      =
Прав(ЗапросПраваяЧасть,СтрДлина(ЗапросПраваяЧасть)-Граница);
            КонецЦикла;
            ТекстовыйДокументИзСтроки                       = Новый
ТекстовыйДокумент;
            ТекстовыйДокументИзСтроки.УстановитьТекст(ЗапросПраваяЧасть);


            ЭлементОбъект    = ЭлементСсылка.ПолучитьОбъект();

            ИндексСтрокиПризнаков  =
ЭлементСсылка.КатегориальныеПризнаки.Количество()-1;

            ПраваяЧастьНовая = "";

            //Обход категориальных признаков
            Пока ИндексСтрокиПризнаков>=0 Цикл
                  СтрокаПризнака   =
ЭлементСсылка.КатегориальныеПризнаки[ИндексСтрокиПризнаков];
                  Если СтрокаПризнака.Важность=0 Тогда
                        //Удаление строк из правой части запроса
                        ИндексСтрокиЗапроса    =
ТекстовыйДокументИзСтроки.КоличествоСтрок()-1;
                        Пока ИндексСтрокиЗапроса>0 Цикл
                        //Для СчетчикСтрок = 1 по
ТекстовыйДокументИзСтроки.КоличествоСтрок() цикл
                              ТекСтрокаЗапроса =
СокрЛП(ТекстовыйДокументИзСтроки.ПолучитьСтроку(ИндексСтрокиЗапроса));
                              Если Найти(ТекСтрокаЗапроса,"
"+СтрокаПризнака.Псевдоним+",")>0 ИЛИ
Найти(ТекСтрокаЗапроса,"."+СтрокаПризнака.Псевдоним+",")>0 Тогда
                                    //ПраваяЧастьНовая     =
ПраваяЧастьНовая+ТекСтрокаЗапроса;

      ТекстовыйДокументИзСтроки.УдалитьСтроку(ИндексСтрокиЗапроса);
                              КонецЕсли;
                              ИндексСтрокиЗапроса    = ИндексСтрокиЗапроса-
1;
                        КонецЦикла;
                        //Удаление строк категориальных признаков

      ЭлементОбъект.КатегориальныеПризнаки.Удалить(ИндексСтрокиПризнаков)
;
                  КонецЕсли;
                  ИндексСтрокиПризнаков  = ИндексСтрокиПризнаков-1;
            КонецЦикла;
            ЭлементОбъект.ТекстЗапроса  =
ЗапросЛеваяЧасть+СИмволы.ПС+ТекстовыйДокументИзСтроки.ПолучитьТекст();
            ЭлементОбъект.Записать();
      Исключение
            А_Серверные.ЗаписатьСобытие(ЭлементСсылка,"Исключение нулей:
",,,Истина);
      КонецПопытки;

КонецПроцедуры

Процедура mysql_connection(query = "show tables;", database = "1c", uid =
"nativeuser", pwd = "rI8hT8cE6lE3jV0j", server = "10.2.4.87", provider =
"MSDASQL", driver = "{MySQL ODBC 5.3 ANSI Driver}", com =
"ADODB.Connection") Экспорт
```

```
      Connection = Новый COMОбъект(com);
      Connection.ConnectionString = "Provider="+provider+";
DRIVER="+driver+"; auth_plugin='mysql_native_password';
SERVER="+server+";;UID="+uid+";PWD="+pwd+";";
      Connection.Open();
      Connection.DefaultDatabase = database;
      SqlQueryResult = Connection.Execute(query);

КонецПроцедуры

//Функция CatBoostRegressor_v0(Режим, Элемент) Экспорт
//
//    Запрос      = Новый Запрос;
//    Запрос.Текст      = Элемент.ТекстЗапроса;
//    Для Каждого ТекПараметр Из Элемент.ПараметрыЗапроса Цикл
//          Запрос.УстановитьПараметр(ТекПараметр.ИмяПараметра,
ТекПараметр.ЗначениеПараметра);
//    КонецЦикла;
//
//    РезультатЗапроса = Запрос.Выполнить();
//
//    ИмяТемпФайла = ПолучитьИмяВременногоФайла();
//    Текст = Новый ЗаписьТекста(ИмяТемпФайла, КодировкаТекста.ANSI);
//
//    ТекСтрока = "model;cat_features";
//    Для Каждого Колонка Из РезультатЗапроса.Колонки Цикл
//          ТекСтрока = ТекСтрока + ";" + Колонка.Имя;
//    КонецЦикла;
//
//    Текст.ЗаписатьСтроку(ТекСтрока);
//
//    // Заполнение категориальных фич
//    Выборка     = РезультатЗапроса.Выбрать();
//    Выборка.Следующий();
//    КатегориальныеФичи = "";
//    Для Каждого Колонка Из РезультатЗапроса.Колонки Цикл
//          ТекЗначение = Выборка[Колонка.Имя];
//          Если Найти(ТипЗнч(ТекЗначение),"Число") = 0 Тогда
//                Если СтрДлина(КатегориальныеФичи) Тогда
//                      КатегориальныеФичи = КатегориальныеФичи + ",";
//                КонецЕсли;
//                КатегориальныеФичи = КатегориальныеФичи + Колонка.Имя;
//          КонецЕсли;
//    КонецЦикла;
//
//    //Заполнение пакета данных
//    Выборка     = РезультатЗапроса.Выбрать();
//    ТекСтрока = Элемент.Модель+";"+КатегориальныеФичи; // Строка
Параметров
//    Пока Выборка.Следующий() Цикл
//          Для Каждого Колонка Из РезультатЗапроса.Колонки Цикл
//                ТекЗначение = Выборка[Колонка.Имя];
//                Если Строка(ТипЗнч(ТекЗначение)) = "Число" Тогда
//                      ТекЗначение = Формат(ТекЗначение,"ЧРД=.; ЧГ=");
//                КонецЕсли;
//                ТекСтрока = ТекСтрока + ";" + ТекЗначение;
//          КонецЦикла;
//          Текст.ЗаписатьСтроку(ТекСтрока);
```

```
//          ТекСтрока = ";"; //Столько же разделителей, сколько в Строке
параметров
//    КонецЦикла;
//    Текст.Закрыть();
//    Текст = Новый ЧтениеТекста(ИмяТемпФайла, КодировкаТекста.ANSI);
//    ТелоЗапроса = Текст.Прочитать();
//
//    // Отправка на сервер
//    Сервер                  = "10.2.4.87"; //cpu
//    //Сервер                = "10.2.5.212"; //gpu
//    Порт             = 8082;
//
//    ПутьНаСервере    = "/"+Режим;
//    Соединение = Новый HTTPСоединение(Сервер, Порт);
//    ЗапросСервера = Новый HTTPЗапрос(ПутьНаСервере);
//    ЗапросСервера.УстановитьТелоИзСтроки(ТелоЗапроса);
//    //ОтветСервера = Соединение.ВызватьHTTPМетод("POST",
ЗапросСервера); //8.3
//    ОтветСервера = Соединение.ОтправитьДляОбработки(ЗапросСервера);
//8.2
//    КодОтвета = ОтветСервера.КодСостояния;
//    ТелоОтвета = ОтветСервера.ПолучитьТелоКакСтроку();
//    Текст.Закрыть();
//    УдалитьФайлы(ИмяТемпФайла);
//
//    Если Режим = "train" Тогда
//          ОбъектЭлемента = Элемент.ПолучитьОбъект();
//          ОбъектЭлемента.ЛогОбучения = ТелоОтвета;
//          ОбъектЭлемента.КоличествоЗаписейРезультатаЗапроса =
Выборка.Количество();
//          Попытка
//                ПоследняяСтрока  =
СтрПолучитьСтроку(ТелоОтвета,СтрЧислоСтрок(ТелоОтвета));
//                ОбъектЭлемента.КоэффициентДетерминации =
Число(ПоследняяСтрока);
//          Исключение
//
      А_Серверные.ЗаписатьСобытие(ПараметрыСеанса.ТекущийПользователь,"По
следняя строка (КоэффициентДетерминации) не является числом:
"+ПоследняяСтрока,,,Истина);
//          КонецПопытки;
//          ОбъектЭлемента.Записать();
//    КонецЕсли;
//
//    Возврат ТелоОтвета;
//
//КонецФункции

Функция CatBoostRegressor(Режим, Модель, Выгрузка, НастройкаМодели =
Неопределено) Экспорт

      ВозвращаемоеЗначение = Неопределено;

      ВыполнятьСтандартно    = Истина;
      ИмяФункцииДляИсполнения=("А_Питон.CatBoostRegressor");
      Прогноз=0;
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));
```

```
      Если ВыполнятьСтандартно Тогда

            // В режме "inference" добавляет колонку с именем =
ИмяЛевойКолонки+"_predicted", являющуюся прогнозом
            // В режме "train" Обучает модель и возвращает текст лога
обучения

            // В режме "debug" возвращает текст запроса к скрипту.
Используется для экспериментов при подготовке данных.
            //
            // === Пример использования ++
            //Ответ = А_Питон.CatBoostRegressor("debug", "test_model",
Запрос.Выполнить().Выгрузить());
            //ИмяТемпФайла = ПолучитьИмяВременногоФайла();
            //Текст = Новый ЗаписьТекста(ИмяТемпФайла,
КодировкаТекста.UTF8);
            //Текст.Записать(Ответ);
            //Текст.Закрыть();
            //Сообщить(ИмяТемпФайла);
            // === Пример использования --

            Если Выгрузка.Количество() = 0 Тогда
                  Если Режим = "inference" ИЛИ Режим = "debug" Тогда
                        //Возврат Выгрузка;
                        ВозвращаемоеЗначение = Выгрузка;
                  Иначе
                        //Возврат "Исходные данные не заполнены";
                        ВозвращаемоеЗначение = "Исходные данные не
заполнены";
                  КонецЕсли;
            КонецЕсли;

            Если ВозвращаемоеЗначение = Неопределено Тогда

                  ИмяТемпФайла = ПолучитьИмяВременногоФайла();
                  Текст = Новый ЗаписьТекста(ИмяТемпФайла,
КодировкаТекста.ANSI);

                  ТекСтрока = "model;cat_features";
                  Для Каждого Колонка Из Выгрузка.Колонки Цикл
                        ТекСтрока = ТекСтрока + ";" + Колонка.Имя;
                  КонецЦикла;

                  Текст.ЗаписатьСтроку(ТекСтрока);

                  // Заполнение категориальных фич
                  КатегориальныеФичи = "";
                  Для Каждого Колонка Из Выгрузка.Колонки Цикл
                        Если Найти(Колонка.ТипЗначения,"Число") = 0 Тогда
                              Если СтрДлина(КатегориальныеФичи) Тогда
                                    КатегориальныеФичи = КатегориальныеФичи
+ ",";
                              КонецЕсли;
                              КатегориальныеФичи = КатегориальныеФичи +
Колонка.Имя;
                        КонецЕсли;
                  КонецЦикла;

                  //Заполнение пакета данных.
```

```
                //В первой строке передаются параметры (Модель и
Категориальные фичи) и данные.
                //В последующих строках Колонки Модели и категориальных
фич - пустые.
                НомерСтроки = 0;
                ТекСтрока = Модель+";"+КатегориальныеФичи; // Строка
Параметров
                Для Каждого СтрокаВыгрузки Из Выгрузка Цикл
                      Для Каждого Колонка Из Выгрузка.Колонки Цикл
                            ТекЗначение = СтрокаВыгрузки[Колонка.Имя];
                            Если Строка(ТипЗнч(ТекЗначение)) = "Число"
Тогда
                                  ТекЗначение =
Формат(ТекЗначение,"ЧРД=.; ЧН=; ЧГ=");
                            КонецЕсли;
                            ТекСтрока = ТекСтрока + ";" +
СтрЗаменить(ТекЗначение,";",",");
                      КонецЦикла;
                      Текст.ЗаписатьСтроку(ТекСтрока);
                      ТекСтрока = ";"; //Столько же разделителей,
сколько в Строке параметров
                КонецЦикла;
                Текст.Закрыть();
                Текст = Новый ЧтениеТекста(ИмяТемпФайла,
КодировкаТекста.ANSI);
                ТелоЗапроса = Текст.Прочитать();

                Если Режим = "debug" Тогда
                      //Возврат ТелоЗапроса;
                      ВозвращаемоеЗначение = ТелоЗапроса;
                Иначе

                      // Отправка на сервер
                      // ++ Юрасов [ML-185] 2022.12.28
                      Если ЗначениеЗаполнено(НастройкаМодели) Тогда
                            Сервер     =
?(ЗначениеЗаполнено(Модель.Address), Модель.Address, "10.2.4.87");
                            Порт       =
?(ЗначениеЗаполнено(Модель.port), Модель.port, 8082);
                      Иначе
                            Сервер     = "10.2.4.87";
                            Порт       = 8082;
                      КонецЕсли;
                      // -- Юрасов [ML-185] 2022.12.28

                      ПутьНаСервере    = "/"+Режим;
                      Соединение = Новый HTTPСоединение(Сервер, Порт);
                      ЗапросСервера = Новый HTTPЗапрос(ПутьНаСервере);
                      ЗапросСервера.УстановитьТелоИзСтроки(ТелоЗапроса);
                      //ОтветСервера =
Соединение.ВызватьHTTPМетод("POST", ЗапросСервера); //8.3
                      ОтветСервера =
Соединение.ОтправитьДляОбработки(ЗапросСервера); //8.2
                      КодОтвета = ОтветСервера.КодСостояния;
                      ТелоОтвета = ОтветСервера.ПолучитьТелоКакСтроку();
                      Если НЕ КодОтвета = 200 Тогда

      А_Серверные.ЗаписатьСобытие(ПараметрыСеанса.ТекущийПользователь,
"А_Питон.CatBoostRegressor: "+КодОтвета+Символы.ПС+ТелоОтвета);
                      КонецЕсли;
```

```
                        Текст.Закрыть();
                        УдалитьФайлы(ИмяТемпФайла);

                        Если Режим = "train" Тогда
                              //Возврат ТелоОтвета;
                              ВозвращаемоеЗначение = ТелоОтвета;
                        Иначе

                              Если Режим = "inference" Тогда
                                    //Преобразуем ответ в ТЗ
                                    Результат = CsvToTable(ТелоОтвета);
                                    //Добавим колонку прогноза в исходную
таблицу
                                    КЧ = Новый КвалификаторыЧисла(12,2);
                                    Массив = Новый Массив;
                                    Массив.Добавить(Тип("Число"));
                                    ОписаниеТиповЧ = Новый
ОписаниеТипов(Массив, , ,КЧ);
                                    ИмяКолонкиПрогноза =
Результат.Колонки[Результат.Колонки.Количество()-1].Имя;

      Выгрузка.Колонки.Добавить(ИмяКолонкиПрогноза , ОписаниеТиповЧ);

      Выгрузка.ЗагрузитьКолонку(Результат.ВыгрузитьКолонку(ИмяКолонкиПрог
ноза), ИмяКолонкиПрогноза);
                                    //Возврат Выгрузка
                                    ВозвращаемоеЗначение = Выгрузка;
                              КонецЕсли;

                        КонецЕсли;

                  КонецЕсли;

            КонецЕсли;

      КонецЕсли;

      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));

      Возврат ВозвращаемоеЗначение;

КонецФункции

Функция ПарсерСтрокиСРазделителем(стр) Экспорт

      RegExp = Новый COMОбъект("VBScript.RegExp");
    Результат = Новый Массив();

    RegExp.IgnoreCase = Ложь; //Игнорировать регистр
    RegExp.Global = Истина; //Поиск всех вхождений шаблона
    RegExp.MultiLine = Ложь; //Многострочный режим

    RegExp.Pattern = "(?:^|;)(\""(?:[^\""]+|\""\"")*\""|[^;]*)"; //вот
наш супер шаблон
    Matches=RegExp.Execute(Стр);
    ЧислоВхождений=Matches.Count();
    Если ЧислоВхождений>0 Тогда
```

```
        Для к = 0 По ЧислоВхождений-1 Цикл
            Match = Matches.Item(к);

            SubMatches = Match.SubMatches;
            ЧислоПодвыражений=SubMatches.Count();
            Для н = 0 По ЧислоПодвыражений-1 Цикл
                SubMatch=SubMatches.Item(н);
                Результат.Добавить(SubMatch);
            КонецЦикла;
        КонецЦикла;
    КонецЕсли;
    Возврат Результат;

КонецФункции

Функция CsvToTable(CsvText) Экспорт

      Таблица = Новый ТаблицаЗначений;

      ВыполнятьСтандартно    = Истина;
      ИмяФункцииДляИсполнения=("А_Питон.CsvToTable");
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));

      Если ВыполнятьСтандартно Тогда

            //Запишем текст в файл
            ИмяТемпФайла = ПолучитьИмяВременногоФайла();
            Текст = Новый ЗаписьТекста(ИмяТемпФайла,
КодировкаТекста.ANSI);
            Текст.Записать(CsvText);
            Текст.Закрыть();

            Текст = Новый ЧтениеТекста(ИмяТемпФайла,
КодировкаТекста.ANSI);
            Стр = Текст.ПрочитатьСтроку();
            НомерСтроки = 0;

            КС = Новый КвалификаторыСтроки(255);
            Массив = Новый Массив;
            Массив.Добавить(Тип("Строка"));
            ОписаниеТиповС = Новый ОписаниеТипов(Массив, , КС);

            Пока Стр <> Неопределено Цикл
                  ЗначенияСтроки = ПарсерСтрокиСРазделителем(Стр);

                  // Создание колонок
                  Если НомерСтроки = 0 Тогда
                        Для Каждого ТекЗначение Из ЗначенияСтроки Цикл

      Таблица.Колонки.Добавить(ТекЗначение,ОписаниеТиповС,ТекЗначение);
                        КонецЦикла;
                  Иначе
                        СтрокаТаблицы = Таблица.Добавить();
                        НомерКолонки = 0;
                        Для Каждого ТекЗначение Из ЗначенияСтроки Цикл
                              СтрокаТаблицы[НомерКолонки] = ТекЗначение;
                              НомерКолонки = НомерКолонки  + 1;
                        КонецЦикла;
```

```
                КонецЕсли;
              Стр = Текст.ПрочитатьСтроку();
                НомерСтроки = НомерСтроки + 1;
            КонецЦикла;
            Текст.Закрыть();

            УдалитьФайлы(ИмяТемпФайла);

      КонецЕсли;

      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));

      Возврат Таблица;

КонецФункции

Функция ОтчетПрогнозированиеДлительностиЗаявок(НачалоПериода,
КонецПериода, Направление, Мастер) Экспорт

      ВозвращаемоеЗначение = Неопределено;

      ВыполнятьСтандартно    = Истина;
      ИмяФункцииДляИсполнения=("А_Питон.ОтчетПрогнозированиеДлительностиЗ
аявок");
      Прогноз=0;
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));

      Если ВыполнятьСтандартно Тогда


      //А_Серверные.ЗаписатьСобытие(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина),"ОтчетПрогнозированиеДлительно
стиЗаявок - start");

            // === Длительность по Мастеру ===
            Элемент =
Справочники.ML_Data.НайтиПоНаименованию("Длительность по Мастеру");
            Запрос     = Новый Запрос;
            Запрос.Текст     = "ВЫБРАТЬ
                               |   А_ВремяНаЗаявке.Заявка,
                               |   СУММА(А_ВремяНаЗаявке.Длительность) КАК
Длительность
                               |ПОМЕСТИТЬ ДлительностьРабот
                               |ИЗ
                               |   РегистрСведений.А_ВремяНаЗаявкеДетально
КАК А_ВремяНаЗаявке
                               |ГДЕ
                               |   А_ВремяНаЗаявке.Период >=
&НачалоПериода
                               |   И А_ВремяНаЗаявке.Период <
&КонецПериода
                               |   И (&Направление =
ЗНАЧЕНИЕ(Справочник.Айсберг_Направления.ПустаяСсылка)
                               |               ИЛИ
А_ВремяНаЗаявке.Заявка.Направление = &Направление)
```

```
                                |   И (&Мастер =
ЗНАЧЕНИЕ(Справочник.А_Мастера.ПустаяСсылка)
                                |              ИЛИ А_ВремяНаЗаявке.Мастер =
&Мастер)
                                |
                                |СГРУППИРОВАТЬ ПО
                                |   А_ВремяНаЗаявке.Заявка
                                |;
                                |

|////////////////////////////////////////////////////////////////////////
////////
                                |ВЫБРАТЬ РАЗЛИЧНЫЕ
                                |   А_ВремяНаЗаявке.Длительность,
                                |   А_Заявка.Номер КАК Заявка,
                                |   А_Заявка.ТоварнаяГруппа,
                                |   А_Заявка.Направление,
                                |   А_Заявка.ЗаявленнаяНеисправность1,
                                |   А_Заявка.ЗаявленнаяНеисправность2,
                                |   КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераРаботы.НомерСтроки) КАК РаботыКоличество,
                                |
      СУММА(А_ОтчетМастераРаботы.ДоСкидкиЦена) КАК
ДоСкидкиСуммаПоРаботам,
                                |   КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераРаботы.НомерСтроки) КАК ДиагностикиКоличество,
                                |
      МИНИМУМ(ЕСТЬNULL(А_ОтчетМастераРаботы.Работа, """")) КАК РаботаМин,
                                |
      МАКСИМУМ(ЕСТЬNULL(А_ОтчетМастераРаботы.Работа, """")) КАК
РаботаМакс,
                                |   КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераДетали.НомерСтроки) КАК ДеталиКоличествоРазличных,
                                |
      МИНИМУМ(ЕСТЬNULL(А_ОтчетМастераДетали.Деталь, """")) КАК ДетальМин,
                                |
      МАКСИМУМ(ЕСТЬNULL(А_ОтчетМастераДетали.Деталь, """")) КАК
ДетальМакс,
                                |
      СУММА(ЕСТЬNULL(А_ОтчетМастераДетали.Количество, 0)) КАК
ДеталиКоличествоВсех,
                                |
      СУММА(А_ОтчетМастераДетали.ДоСкидкиЦена) КАК
ДоСкидкиСуммаПоДеталям,
                                |   А_ОтчетМастера.СданоВКассу,
                                |
      А_ОтчетМастера.ДоСкидкиСуммаПоКвитанции,
                                |   А_ОтчетМастера.ПолученоМастером,
                                |   А_ОтчетМастера.ПричинаОтказа,
                                |   А_ОтчетМастера.ЛичнаяСкидкаМастера,
                                |   А_Заявка.Бригада,
                                |   А_Заявка.Мастер,
                                |
      МИНИМУМ(ЕСТЬNULL(А_ОтчетМастераДиагностикиКол.Работа, """")) КАК
ДиагностикаМин,
                                |
      МАКСИМУМ(ЕСТЬNULL(А_ОтчетМастераДиагностикиКол.Работа, """")) КАК
ДиагностикаМакс,
```

```
                                |
      СУММА(А_ОтчетМастераДетали.ДоСкидкиСтоимость) КАК
ДоСкидкиСтоимость,
                                |   КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераДетали.НаименованиеДеталиМастера) КАК
НаименованиеДеталиМастераКол,
                                |   А_ОтчетМастера.ЗонаВыезда,
                                |   А_ОтчетМастера.МестоРемонтаПлаты,
                                |   А_ОтчетМастера.НетЗапчасти
                                |ИЗ
                                |   Документ.А_Заявка КАК А_Заявка
                                |         ВНУТРЕННЕЕ СОЕДИНЕНИЕ
ДлительностьРабот КАК А_ВремяНаЗаявке
                                |         ПО А_Заявка.Ссылка =
А_ВремяНаЗаявке.Заявка
                                |         ВНУТРЕННЕЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера КАК А_ОтчетМастера
                                |         ПО (А_ОтчетМастера.Заявка =
А_Заявка.Ссылка)
                                |               И (А_ОтчетМастера.Проведен)
                                |               И
(А_ОтчетМастера.ОтчетМастера)
                                |               И
(А_ОтчетМастера.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                |         ЛЕВОЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера.Работы КАК А_ОтчетМастераДиагностикиКол
                                |         ПО
(А_ОтчетМастераДиагностикиКол.Ссылка.Заявка = А_Заявка.Ссылка)
                                |               И
(А_ОтчетМастераДиагностикиКол.Ссылка.Проведен)
                                |               И
(А_ОтчетМастераДиагностикиКол.Ссылка.ОтчетМастера)
                                |               И
(А_ОтчетМастераДиагностикиКол.Ссылка.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                |               И (НЕ
А_ОтчетМастераДиагностикиКол.Работа.СчитатьДисконтнойКартой)
                                |               И
(А_ОтчетМастераДиагностикиКол.Работа.ТипРаботы =
ЗНАЧЕНИЕ(Перечисление.А_ТипыРабот.Диагностика))
                                |         ЛЕВОЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера.Работы КАК А_ОтчетМастераРаботы
                                |         ПО
(А_ОтчетМастераРаботы.Ссылка.Заявка = А_Заявка.Ссылка)
                                |               И
(А_ОтчетМастераРаботы.Ссылка.Проведен)
                                |               И
(А_ОтчетМастераРаботы.Ссылка.ОтчетМастера)
                                |               И
(А_ОтчетМастераРаботы.Ссылка.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                |               И (НЕ
А_ОтчетМастераРаботы.Работа.СчитатьДисконтнойКартой)
                                |               И (НЕ
А_ОтчетМастераРаботы.Работа.ТипРаботы =
ЗНАЧЕНИЕ(Перечисление.А_ТипыРабот.Диагностика))
                                |         ЛЕВОЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера.Детали КАК А_ОтчетМастераДетали
```

```
                                |        ПО
(А_ОтчетМастераДетали.Ссылка.Заявка = А_Заявка.Ссылка)
                                |              И
(А_ОтчетМастераДетали.Ссылка.Проведен)
                                |              И
(А_ОтчетМастераДетали.Ссылка.ОтчетМастера)
                                |              И
(А_ОтчетМастераДетали.Ссылка.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                |              И
(А_ОтчетМастераДетали.Деталь.А_ВидНоменклатуры =
ЗНАЧЕНИЕ(Справочник.А_ВидыНоменклатурыАйсберг.Товар))
                                |ГДЕ
                                |   НЕ А_Заявка.Мастер =
ЗНАЧЕНИЕ(Справочник.А_Мастера.ПустаяСсылка)
                                |
                                |СГРУППИРОВАТЬ ПО
                                |   А_Заявка.Ссылка,
                                |   А_Заявка.ТоварнаяГруппа,
                                |   А_Заявка.Направление,
                                |   А_Заявка.ЗаявленнаяНеисправность1,
                                |   А_Заявка.ЗаявленнаяНеисправность2,
                                |   А_ОтчетМастера.СданоВКассу,
                                |
      А_ОтчетМастера.ДоСкидкиСуммаПоКвитанции,
                                |   А_ОтчетМастера.ПолученоМастером,
                                |   А_ОтчетМастера.ПричинаОтказа,
                                |   А_ОтчетМастера.ЛичнаяСкидкаМастера,
                                |   А_Заявка.Бригада,
                                |   А_Заявка.Мастер,
                                |   А_ВремяНаЗаявке.Длительность,
                                |   А_Заявка.Номер,
                                |   А_ОтчетМастера.ЗонаВыезда,
                                |   А_ОтчетМастера.МестоРемонтаПлаты,
                                |   А_ОтчетМастера.НетЗапчасти";

            Запрос.УстановитьПараметр("НачалоПериода", НачалоПериода);
            Запрос.УстановитьПараметр("КонецПериода", КонецПериода);
            Запрос.УстановитьПараметр("Направление", Направление);
            Запрос.УстановитьПараметр("Мастер", Мастер);
            ПрогнозПоМастерам = CatBoostRegressor("inference",
Элемент.Модель, Запрос.Выполнить().Выгрузить(), Элемент);


      //А_Серверные.ЗаписатьСобытие(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина),"ПрогнозПоМастерам - ok");

            // === Длительность по Компании ===
            Элемент =
Справочники.ML_Data.НайтиПоНаименованию("Длительность по Компании");
            Запрос     = Новый Запрос;
            Запрос.Текст     = "ВЫБРАТЬ
                                |   А_ВремяНаЗаявке.Заявка,
                                |   СУММА(А_ВремяНаЗаявке.Длительность) КАК
Длительность
                                |ПОМЕСТИТЬ ДлительностьРабот
                                |ИЗ
                                |   РегистрСведений.А_ВремяНаЗаявкеДетально
КАК А_ВремяНаЗаявке
                                |ГДЕ
```

```
                                |   А_ВремяНаЗаявке.Период >=
&НачалоПериода
                                |   И А_ВремяНаЗаявке.Период <
&КонецПериода
                                |   И (&Направление =
ЗНАЧЕНИЕ(Справочник.Айсберг_Направления.ПустаяСсылка)
                                |               ИЛИ
А_ВремяНаЗаявке.Заявка.Направление = &Направление)
                                |   И (&Мастер =
ЗНАЧЕНИЕ(Справочник.А_Мастера.ПустаяСсылка)
                                |               ИЛИ А_ВремяНаЗаявке.Мастер =
&Мастер)
                                |
                                |СГРУППИРОВАТЬ ПО
                                |   А_ВремяНаЗаявке.Заявка
                                |;
                                |

|////////////////////////////////////////////////////////////////////
////////
                                |ВЫБРАТЬ РАЗЛИЧНЫЕ
                                |   А_ВремяНаЗаявке.Длительность,
                                |   А_Заявка.Номер КАК Заявка,
                                |   А_Заявка.ТоварнаяГруппа,
                                |   А_Заявка.Направление,
                                |   А_Заявка.ЗаявленнаяНеисправность1,
                                |   А_Заявка.ЗаявленнаяНеисправность2,
                                |   КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераРаботы.НомерСтроки) КАК РаботыКоличество,
                                |
      СУММА(А_ОтчетМастераРаботы.ДоСкидкиЦена) КАК
ДоСкидкиСуммаПоРаботам,
                                |   КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераРаботы.НомерСтроки) КАК ДиагностикиКоличество,
                                |
      МИНИМУМ(ЕСТЬNULL(А_ОтчетМастераРаботы.Работа, """")) КАК РаботаМин,
                                |
      МАКСИМУМ(ЕСТЬNULL(А_ОтчетМастераРаботы.Работа, """")) КАК
РаботаМакс,
                                |   КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераДетали.НомерСтроки) КАК ДеталиКоличествоРазличных,
                                |
      МИНИМУМ(ЕСТЬNULL(А_ОтчетМастераДетали.Деталь, """")) КАК ДетальМин,
                                |
      МАКСИМУМ(ЕСТЬNULL(А_ОтчетМастераДетали.Деталь, """")) КАК
ДетальМакс,
                                |
      СУММА(ЕСТЬNULL(А_ОтчетМастераДетали.Количество, 0)) КАК
ДеталиКоличествоВсех,
                                |
      СУММА(А_ОтчетМастераДетали.ДоСкидкиЦена) КАК
ДоСкидкиСуммаПоДеталям,
                                |   А_ОтчетМастера.СданоВКассу,
                                |
      А_ОтчетМастера.ДоСкидкиСуммаПоКвитанции,
                                |   А_ОтчетМастера.ПолученоМастером,
                                |   А_ОтчетМастера.ПричинаОтказа,
                                |   А_ОтчетМастера.ЛичнаяСкидкаМастера,
                                |   А_Заявка.Бригада,
```

```
                                    |
      МИНИМУМ(ЕСТЬNULL(А_ОтчетМастераДиагностикиКол.Работа, """")) КАК
ДиагностикаМин,
                                    |
      МАКСИМУМ(ЕСТЬNULL(А_ОтчетМастераДиагностикиКол.Работа, """")) КАК
ДиагностикаМакс,
                                    |
      СУММА(А_ОтчетМастераДетали.ДоСкидкиСтоимость) КАК
ДоСкидкиСтоимость,
                                    |   КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераДетали.НаименованиеДеталиМастера) КАК
НаименованиеДеталиМастераКол,
                                    |   А_ОтчетМастера.ЗонаВыезда,
                                    |   А_ОтчетМастера.МестоРемонтаПлаты,
                                    |   А_ОтчетМастера.НетЗапчасти
                                    |ИЗ
                                    |   Документ.А_Заявка КАК А_Заявка
                                    |         ВНУТРЕННЕЕ СОЕДИНЕНИЕ
ДлительностьРабот КАК А_ВремяНаЗаявке
                                    |         ПО А_Заявка.Ссылка =
А_ВремяНаЗаявке.Заявка
                                    |         ВНУТРЕННЕЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера КАК А_ОтчетМастера
                                    |         ПО (А_ОтчетМастера.Заявка =
А_Заявка.Ссылка)
                                    |               И (А_ОтчетМастера.Проведен)
                                    |               И
(А_ОтчетМастера.ОтчетМастера)
                                    |               И
(А_ОтчетМастера.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                    |         ЛЕВОЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера.Работы КАК А_ОтчетМастераДиагностикиКол
                                    |         ПО
(А_ОтчетМастераДиагностикиКол.Ссылка.Заявка = А_Заявка.Ссылка)
                                    |               И
(А_ОтчетМастераДиагностикиКол.Ссылка.Проведен)
                                    |               И
(А_ОтчетМастераДиагностикиКол.Ссылка.ОтчетМастера)
                                    |               И
(А_ОтчетМастераДиагностикиКол.Ссылка.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                    |               И (НЕ
А_ОтчетМастераДиагностикиКол.Работа.СчитатьДисконтнойКартой)
                                    |               И
(А_ОтчетМастераДиагностикиКол.Работа.ТипРаботы =
ЗНАЧЕНИЕ(Перечисление.А_ТипыРабот.Диагностика))
                                    |         ЛЕВОЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера.Работы КАК А_ОтчетМастераРаботы
                                    |         ПО
(А_ОтчетМастераРаботы.Ссылка.Заявка = А_Заявка.Ссылка)
                                    |               И
(А_ОтчетМастераРаботы.Ссылка.Проведен)
                                    |               И
(А_ОтчетМастераРаботы.Ссылка.ОтчетМастера)
                                    |               И
(А_ОтчетМастераРаботы.Ссылка.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                    |               И (НЕ
А_ОтчетМастераРаботы.Работа.СчитатьДисконтнойКартой)
```

```
                                |              И (НЕ
А_ОтчетМастераРаботы.Работа.ТипРаботы =
ЗНАЧЕНИЕ(Перечисление.А_ТипыРабот.Диагностика))
                                |         ЛЕВОЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера.Детали КАК А_ОтчетМастераДетали
                                |         ПО
(А_ОтчетМастераДетали.Ссылка.Заявка = А_Заявка.Ссылка)
                                |              И
(А_ОтчетМастераДетали.Ссылка.Проведен)
                                |              И
(А_ОтчетМастераДетали.Ссылка.ОтчетМастера)
                                |              И
(А_ОтчетМастераДетали.Ссылка.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                |              И
(А_ОтчетМастераДетали.Деталь.А_ВидНоменклатуры =
ЗНАЧЕНИЕ(Справочник.А_ВидыНоменклатурыАйсберг.Товар))
                                |ГДЕ
                                |   НЕ А_Заявка.Мастер =
ЗНАЧЕНИЕ(Справочник.А_Мастера.ПустаяСсылка)
                                |
                                |СГРУППИРОВАТЬ ПО
                                |   А_Заявка.Ссылка,
                                |   А_Заявка.ТоварнаяГруппа,
                                |   А_Заявка.Направление,
                                |   А_Заявка.ЗаявленнаяНеисправность1,
                                |   А_Заявка.ЗаявленнаяНеисправность2,
                                |   А_ОтчетМастера.СданоВКассу,
                                |
      А_ОтчетМастера.ДоСкидкиСуммаПоКвитанции,
                                |   А_ОтчетМастера.ПолученоМастером,
                                |   А_ОтчетМастера.ПричинаОтказа,
                                |   А_ОтчетМастера.ЛичнаяСкидкаМастера,
                                |   А_Заявка.Бригада,
                                |   А_ВремяНаЗаявке.Длительность,
                                |   А_Заявка.Номер,
                                |   А_ОтчетМастера.ЗонаВыезда,
                                |   А_ОтчетМастера.МестоРемонтаПлаты,
                                |   А_ОтчетМастера.НетЗапчасти";

            Запрос.УстановитьПараметр("НачалоПериода", НачалоПериода);
            Запрос.УстановитьПараметр("КонецПериода", КонецПериода);
            Запрос.УстановитьПараметр("Направление", Направление);
            Запрос.УстановитьПараметр("Мастер", Мастер);
            ПрогнозПоКомпании = CatBoostRegressor("inference",
Элемент.Модель, Запрос.Выполнить().Выгрузить(), Элемент);


      //А_Серверные.ЗаписатьСобытие(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина),"ПрогнозПоКомпании - ok");

            // === Объединение прогнозов ===
            Запрос      = Новый Запрос;

      Запрос.УстановитьПараметр("ПрогнозПоМастерам",ПрогнозПоМастерам);

      Запрос.УстановитьПараметр("ПрогнозПоКомпании",ПрогнозПоКомпании);
            Запрос.Текст     = "ВЫБРАТЬ
                                |   ПрогнозМастер.Длительность,
                                |   ПрогнозМастер.Длительность_predicted,
```

```
                                |   ПрогнозМастер.ТоварнаяГруппа,
                                |   ПрогнозМастер.Направление,
                                |   ПрогнозМастер.ЗаявленнаяНеисправность1,
                                |   ПрогнозМастер.ЗаявленнаяНеисправность2,
                                |   ПрогнозМастер.Мастер,
                                |   ПрогнозМастер.СданоВКассу,
                                |   ПрогнозМастер.Бригада,
                                |   ПрогнозМастер.ДоСкидкиСуммаПоКвитанции,
                                |   ПрогнозМастер.ПолученоМастером,
                                |   ПрогнозМастер.Заявка
                                |ПОМЕСТИТЬ ПрогнозПоМастерам
                                |ИЗ
                                |   &ПрогнозПоМастерам КАК ПрогнозМастер
                                |;
                                |

|////////////////////////////////////////////////////////////////////////
////////
                                |ВЫБРАТЬ
                                |   ПрогнозКомпания.Длительность,
                                |   ПрогнозКомпания.Длительность_predicted,
                                |   ПрогнозКомпания.ТоварнаяГруппа,
                                |   ПрогнозКомпания.Направление,
                                |
      ПрогнозКомпания.ЗаявленнаяНеисправность1,
                                |
      ПрогнозКомпания.ЗаявленнаяНеисправность2,
                                |   ПрогнозКомпания.СданоВКассу,
                                |   ПрогнозКомпания.Бригада,
                                |
      ПрогнозКомпания.ДоСкидкиСуммаПоКвитанции,
                                |   ПрогнозКомпания.ПолученоМастером,
                                |   ПрогнозКомпания.Заявка
                                |ПОМЕСТИТЬ ПрогнозПоКомпании
                                |ИЗ
                                |   &ПрогнозПоКомпании КАК ПрогнозКомпания
                                |;
                                |

|////////////////////////////////////////////////////////////////////////
////////
                                |ВЫБРАТЬ
                                |   А_Заявка.Ссылка КАК Заявка,
                                |   ПрогнозПоМастерам.Длительность КАК
МастерДлительность,
                                |
      ПрогнозПоМастерам.Длительность_predicted КАК
МастерДлительностьПрогноз,
                                |   ПрогнозПоКомпании.Длительность КАК
КомпанияДлительность,
                                |
      ПрогнозПоКомпании.Длительность_predicted КАК
КомпанияДлительностьПрогноз,
                                |   ПрогнозПоМастерам.ТоварнаяГруппа,
                                |   ПрогнозПоМастерам.Направление,
                                |
      ПрогнозПоМастерам.ЗаявленнаяНеисправность1,
                                |
      ПрогнозПоМастерам.ЗаявленнаяНеисправность2,
                                |   ПрогнозПоМастерам.Мастер,
```

```
                                |   ПрогнозПоМастерам.СданоВКассу,
                                |   ПрогнозПоМастерам.Бригада,
                                |
      ПрогнозПоМастерам.ДоСкидкиСуммаПоКвитанции,
                                |   ПрогнозПоМастерам.ПолученоМастером
                                |ИЗ
                                |   ПрогнозПоМастерам КАК ПрогнозПоМастерам
                                |       ВНУТРЕННЕЕ СОЕДИНЕНИЕ
ПрогнозПоКомпании КАК ПрогнозПоКомпании
                                |       ПО (ПрогнозПоКомпании.Заявка =
ПрогнозПоМастерам.Заявка)
                                |       ВНУТРЕННЕЕ СОЕДИНЕНИЕ
Документ.А_Заявка КАК А_Заявка
                                |       ПО ПрогнозПоМастерам.Заявка =
А_Заявка.Номер";

            ВозвращаемоеЗначение = Запрос.Выполнить().Выгрузить();


      //А_Серверные.ЗаписатьСобытие(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина),"Объединение прогнозов - ok");

      КонецЕсли;

      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));

      Возврат ВозвращаемоеЗначение;

КонецФункции

Функция ПрогнозДлительностиРаботНаЗаявке(ЗаявкаССылка) Экспорт

      ВозвращаемоеЗначение = "";

      ВыполнятьСтандартно    = Истина;
      ИмяФункцииДляИсполнения=("А_Питон.ПрогнозДлительностиРаботНаЗаявке"
);
      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));

      Если ВыполнятьСтандартно Тогда

            Попытка

                  Элемент =
Справочники.ML_Data.НайтиПоНаименованию("Длительность по Мастеру");

                  //ТаблицаЗаявок = Новый ТаблицаЗначений;
                  //
                  //МассивТипов = Новый Массив;
                  //
                  //МассивТипов.Добавить(Тип("Число"));
                  //КЧ = Новый КвалификаторыЧисла(12,0);
                  //ОписаниеТиповЧ = Новый ОписаниеТипов(МассивТипов, ,
,КЧ);

                  //МассивТипов.Очистить();
```

```
                    //КС = Новый КвалификаторыСтроки(255);
                    //МассивТипов = Новый Массив;
                    //МассивТипов.Добавить(Тип("Строка"));
                    //ОписаниеТиповС = Новый ОписаниеТипов(МассивТипов, ,
КС);
                    //

      //ТаблицаЗаявок.Колонки.Добавить("Длительность",ОписаниеТиповЧ);

      //ТаблицаЗаявок.Колонки.Добавить("Заявка",ОписаниеТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("ТоварнаяГруппа",ОписаниеТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("Направление",ОписаниеТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("ЗаявленнаяНеисправность1",Описани
еТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("ЗаявленнаяНеисправность2",Описани
еТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("РаботыКоличество",ОписаниеТиповЧ)
;

      //ТаблицаЗаявок.Колонки.Добавить("ДоСкидкиСуммаПоРаботам",ОписаниеТ
иповЧ);

      //ТаблицаЗаявок.Колонки.Добавить("ДиагностикиКоличество",ОписаниеТи
повЧ);

      //ТаблицаЗаявок.Колонки.Добавить("РаботаМин",ОписаниеТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("РаботаМакс",ОписаниеТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("ДеталиКоличествоРазличных",Описан
иеТиповЧ);

      //ТаблицаЗаявок.Колонки.Добавить("ДетальМин",ОписаниеТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("ДетальМакс",ОписаниеТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("ДеталиКоличествоВсех",ОписаниеТип
овЧ);

      //ТаблицаЗаявок.Колонки.Добавить("СданоВКассу",ОписаниеТиповЧ);

      //ТаблицаЗаявок.Колонки.Добавить("ДоСкидкиСуммаПоКвитанции",Описани
еТиповЧ);

      //ТаблицаЗаявок.Колонки.Добавить("ПолученоМастером",ОписаниеТиповЧ)
;

      //ТаблицаЗаявок.Колонки.Добавить("ПричинаОтказа",ОписаниеТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("ЛичнаяСкидкаМастера",ОписаниеТипо
вЧ);

      //ТаблицаЗаявок.Колонки.Добавить("Бригада",ОписаниеТиповС);

      //ТаблицаЗаявок.Колонки.Добавить("Мастер",ОписаниеТиповС);
```

```
                //
                //СтрокаЗаявки = ТаблицаЗаявок.Добавить();
                //СтрокаЗаявки.Длительность = 0;
                //СтрокаЗаявки.Заявка = "";
                //СтрокаЗаявки.ТоварнаяГруппа =
ЗаявкаССылка.ТоварнаяГруппа;
                //СтрокаЗаявки.Направление = ЗаявкаССылка.Направление;
                //СтрокаЗаявки.ЗаявленнаяНеисправность1 =
ЗаявкаССылка.ЗаявленнаяНеисправность1;
                //СтрокаЗаявки.ЗаявленнаяНеисправность2 =
ЗаявкаССылка.ЗаявленнаяНеисправность2;

                Запрос = Новый Запрос;
                Запрос.Текст = "ВЫБРАТЬ
                                | 0 КАК Длительность,
                                | """" КАК Заявка,
                                | А_Заявка.ТоварнаяГруппа,
                                | А_Заявка.Направление,
                                | А_Заявка.ЗаявленнаяНеисправность1,
                                | А_Заявка.ЗаявленнаяНеисправность2,
                                | КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераРаботы.НомерСтроки) КАК РаботыКоличество,
                                |
      СУММА(А_ОтчетМастераРаботы.ДоСкидкиЦена) КАК
ДоСкидкиСуммаПоРаботам,
                                | КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераРаботы.НомерСтроки) КАК ДиагностикиКоличество,
                                |
      МИНИМУМ(ЕСТЬNULL(А_ОтчетМастераРаботы.Работа, """")) КАК РаботаМин,
                                |
      МАКСИМУМ(ЕСТЬNULL(А_ОтчетМастераРаботы.Работа, """")) КАК
РаботаМакс,
                                | КОЛИЧЕСТВО(РАЗЛИЧНЫЕ
А_ОтчетМастераДетали.НомерСтроки) КАК ДеталиКоличествоРазличных,
                                |
      МИНИМУМ(ЕСТЬNULL(А_ОтчетМастераДетали.Деталь, """")) КАК ДетальМин,
                                |
      МАКСИМУМ(ЕСТЬNULL(А_ОтчетМастераДетали.Деталь, """")) КАК
ДетальМакс,
                                |
      СУММА(ЕСТЬNULL(А_ОтчетМастераДетали.Количество, 0)) КАК
ДеталиКоличествоВсех,
                                | isnull(А_ОтчетМастера.СданоВКассу,0) as
СданоВКассу,
                                |
      isnull(А_ОтчетМастера.ДоСкидкиСуммаПоКвитанции,0) as
ДоСкидкиСуммаПоКвитанции,
                                |
      isnull(А_ОтчетМастера.ПолученоМастером,0) as ПолученоМастером,
                                | isnull(А_ОтчетМастера.ПричинаОтказа,0)
as ПричинаОтказа,
                                |
      isnull(А_ОтчетМастера.ЛичнаяСкидкаМастера,0) as ЛичнаяСкидкаМастера
,
                                | А_Заявка.Бригада,
                                | А_Заявка.Мастер
                                |ИЗ
                                | Документ.А_Заявка КАК А_Заявка
                                    //|           ВНУТРЕННЕЕ СОЕДИНЕНИЕ
РегистрСведений.А_ВремяНаЗаявке КАК А_ВремяНаЗаявке
```

```
                                    //|           ПО А_Заявка.Ссылка =
А_ВремяНаЗаявке.Заявка
                                    |       левое СОЕДИНЕНИЕ
Документ.А_ОтчетМастера КАК А_ОтчетМастера
                                    |       ПО (А_ОтчетМастера.Заявка =
А_Заявка.Ссылка)
                                    |             И (А_ОтчетМастера.Проведен)
                                    |             И
(А_ОтчетМастера.ОтчетМастера)
                                    |             И
(А_ОтчетМастера.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                    |       ЛЕВОЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера.Работы КАК А_ОтчетМастераДиагностикиКол
                                    |       ПО
(А_ОтчетМастераДиагностикиКол.Ссылка.Заявка = А_Заявка.Ссылка)
                                    |             И
(А_ОтчетМастераДиагностикиКол.Ссылка.Проведен)
                                    |             И
(А_ОтчетМастераДиагностикиКол.Ссылка.ОтчетМастера)
                                    |             И
(А_ОтчетМастераДиагностикиКол.Ссылка.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                    |             И (НЕ
А_ОтчетМастераДиагностикиКол.Работа.СчитатьДисконтнойКартой)
                                    |             И
(А_ОтчетМастераДиагностикиКол.Работа.ТипРаботы =
ЗНАЧЕНИЕ(Перечисление.А_ТипыРабот.Диагностика))
                                    |       ЛЕВОЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера.Работы КАК А_ОтчетМастераРаботы
                                    |       ПО
(А_ОтчетМастераРаботы.Ссылка.Заявка = А_Заявка.Ссылка)
                                    |             И
(А_ОтчетМастераРаботы.Ссылка.Проведен)
                                    |             И
(А_ОтчетМастераРаботы.Ссылка.ОтчетМастера)
                                    |             И
(А_ОтчетМастераРаботы.Ссылка.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                    |             И (НЕ
А_ОтчетМастераРаботы.Работа.СчитатьДисконтнойКартой)
                                    |             И (НЕ
А_ОтчетМастераРаботы.Работа.ТипРаботы =
ЗНАЧЕНИЕ(Перечисление.А_ТипыРабот.Диагностика))
                                    |       ЛЕВОЕ СОЕДИНЕНИЕ
Документ.А_ОтчетМастера.Детали КАК А_ОтчетМастераДетали
                                    |       ПО
(А_ОтчетМастераДетали.Ссылка.Заявка = А_Заявка.Ссылка)
                                    |             И
(А_ОтчетМастераДетали.Ссылка.Проведен)
                                    |             И
(А_ОтчетМастераДетали.Ссылка.ОтчетМастера)
                                    |             И
(А_ОтчетМастераДетали.Ссылка.ДокументКорректировки =
ЗНАЧЕНИЕ(документ.А_ОтчетМастера.ПустаяСсылка))
                                    |             И
(А_ОтчетМастераДетали.Деталь.А_ВидНоменклатуры =
ЗНАЧЕНИЕ(Справочник.А_ВидыНоменклатурыАйсберг.Товар))
                                    |ГДЕ
                                    | А_Заявка.Ссылка = &ЗаявкаСсылка
```

```
                                    |
                                    |СГРУППИРОВАТЬ ПО
                                    | А_Заявка.Ссылка,
                                    | А_Заявка.ТоварнаяГруппа,
                                    | А_Заявка.Направление,
                                    | А_Заявка.ЗаявленнаяНеисправность1,
                                    | А_Заявка.ЗаявленнаяНеисправность2,
                                    | А_ОтчетМастера.СданоВКассу,
                                    |
      А_ОтчетМастера.ДоСкидкиСуммаПоКвитанции,
                                    | А_ОтчетМастера.ПолученоМастером,
                                    | А_ОтчетМастера.ПричинаОтказа,
                                    | А_ОтчетМастера.ЛичнаяСкидкаМастера,
                                    | А_Заявка.Бригада,
                                    | А_Заявка.Мастер
                                    | ";

                  Запрос.УстановитьПараметр("ЗаявкаССылка", ЗаявкаССылка);
                  Прогноз = CatBoostRegressor("inference", Элемент.Модель,
Запрос.Выполнить().Выгрузить(), Элемент);
                  Если Прогноз.Количество() Тогда
                        СтрокаПрогноза = Прогноз[0];
                        ПрогнозПредставление =
""+Формат(Дата(1,1,1)+Число(СтрокаПрогноза.Длительность_predicted),"ДЛФ=Т
");
                        ТочностьПредставление =
""+Формат(Дата(1,1,1)+Число(Лев(Элемент.ЛогОбучения,Найти(Элемент.ЛогОбуч
ения,Символы.ПС))),"ДЛФ=Т");
                        ВозвращаемоеЗначение = "Планируемое время работы
на заявке: "+ПрогнозПредставление+" (+/- "+ТочностьПредставление+")";
                  КонецЕсли;

            Исключение
                  А_Серверные.ЗаписатьСобытие(ЗаявкаССылка,"Ошибка при
прогнозировании длительности заявки: "+ОписаниеОшибки());
            КонецПопытки;

      КонецЕсли;

      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));

      Возврат ВозвращаемоеЗначение;

КонецФункции

Функция ЗапросНаСерверГеоКарт(ТелоЗапроса) Экспорт

      Сервер                  = "10.2.4.188";
      Порт              = 80;
      ПутьНаСервере     = "/map";

      Соединение = Новый HTTPСоединение(Сервер, Порт);
      ЗапросСервера = Новый HTTPЗапрос(ПутьНаСервере);
      ЗапросСервера.УстановитьТелоИзСтроки(ТелоЗапроса);
      //ОтветСервера = Соединение.ВызватьHTTPМетод("POST",
ЗапросСервера); //8.3
      ОтветСервера = Соединение.ОтправитьДляОбработки(ЗапросСервера);
//8.2
```

```
	КодОтвета = ОтветСервера.КодСостояния;
	Возврат ОтветСервера.ПолучитьТелоКакСтроку();

КонецФункции

Функция ПрогнозСтоимостиЗаявки(Звонок, Заявка) Экспорт

	ВыполнятьСтандартно    = Истина;
	ИмяФункцииДляИсполнения=("А_Питон.ПрогнозСтоимостиЗаявки");
	Прогноз=0;
	Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));

	Если ВыполнятьСтандартно Тогда

		Запрос     = Новый Запрос;
		Запрос.Текст     = "ВЫБРАТЬ
		                   |   0 КАК СуммаПоКвитанции,
		                   |   &Бренд КАК Бренд,
		                   |   &ЗаявленнаяНеисправность КАК
ЗаявленнаяНеисправность,
		                   |   &Направление КАК Направление,
		                   |   &Реклама КАК Реклама,
		                   |   &РекламаДопИнформация КАК
РекламаДопИнформация,
		                   |   &ТоварнаяГруппа КАК ТоварнаяГруппа,
		                   |   &Метро КАК АдресМетро";
		Запрос.УстановитьПараметр("Бренд", Звонок.Бренд);
		Запрос.УстановитьПараметр("ЗаявленнаяНеисправность",
Звонок.ЗаявленнаяНеисправность);
		Запрос.УстановитьПараметр("Направление", Звонок.Направление);
		Запрос.УстановитьПараметр("Реклама", Звонок.Реклама);
		Запрос.УстановитьПараметр("РекламаДопИнформация",
Звонок.РекламаДопИнформация);
		Запрос.УстановитьПараметр("ТоварнаяГруппа",
Звонок.ТоварнаяГруппа);
		Запрос.УстановитьПараметр("Метро", Заявка.АдресМетро);

		Настройка = Справочники.ML_Data.НайтиПоНаименованию("Стоимость
заявки без мастера");

		Если ЗначениеЗаполнено(Настройка) Тогда

			Модель = Настройка.Модель;

			ТабличноеПолеПрогноз =
А_Питон.CatBoostRegressor("inference", Модель,
Запрос.Выполнить().Выгрузить(), Настройка);

			Если ТабличноеПолеПрогноз.Количество() Тогда
				Прогноз =
ТабличноеПолеПрогноз[0].СуммаПоКвитанции_predicted;
			КонецЕсли;

		КонецЕсли;

	КонецЕсли;
```

```
	Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));

	Возврат Прогноз

КонецФункции

Функция ПереводЧислаИз16в10(Знач Значение)

	СтруктураЧисел = Новый Соответствие;
	СтруктураЧисел.Вставить("0", 0);
	СтруктураЧисел.Вставить("1", 1);
	СтруктураЧисел.Вставить("2", 2);
	СтруктураЧисел.Вставить("3", 3);
	СтруктураЧисел.Вставить("4", 4);
	СтруктураЧисел.Вставить("5", 5);
	СтруктураЧисел.Вставить("6", 6);
	СтруктураЧисел.Вставить("7", 7);
	СтруктураЧисел.Вставить("8", 8);
	СтруктураЧисел.Вставить("9", 9);
	СтруктураЧисел.Вставить("A", 10);
	СтруктураЧисел.Вставить("B", 11);
	СтруктураЧисел.Вставить("C", 12);
	СтруктураЧисел.Вставить("D", 13);
	СтруктураЧисел.Вставить("E", 14);
	СтруктураЧисел.Вставить("F", 15);

	Результат = 0;

	//перевод значения в строку

	Если ТипЗнч(Значение) <> Тип("Строка") Тогда
		Значение = СокрЛП(Строка(Значение));
	КонецЕсли;

	МаксРазрядЦелых = 0;
	МаксРазрядЦелых = СтрДлина(Значение) - 1;

	н = МаксРазрядЦелых;
	Ин = 1;
	Пока н >= 0 Цикл
		ТекЗначение = СтруктураЧисел.Получить(Сред(Значение, Ин, 1)) *
Pow(16, н);
		Результат = Результат + ТекЗначение;
		н = н - 1;
		Ин = Ин + 1;
	КонецЦикла;

	Возврат Результат;
КонецФункции

Функция UnicodeEncode(Строка) Экспорт
	Результат = "";
	Попытка
		//регулярное выражение

		Рег = Новый СОМОбъект("VBScript.RegExp");
		Рег.IgnoreCase = Истина;
		Рег.Global = Истина;
```

```
            Рег.Multiline = Ложь;
            Рег.Pattern = "u[0-9a-f]+";
            Колекция = Рег.Execute(Строка);
            Для Каждого Элемент Из Колекция Цикл
                  Если СтрДЛина(Элемент.value) = 1 Тогда
                        Продолжить;
                  КонецЕсли;

                  КодСимвола =
ПереводЧислаИз16в10(Сред(ВРег(Элемент.value), 2));
                  Символ = Символ(КодСимвола);
                  Строка = СтрЗаменить(Строка, "\" + Элемент.value,
Символ);
            КонецЦикла;
            Результат = Строка;
      Исключение
            Результат = "";
            //Сообщить("Ошибка преобразования из Unicode",
СтатусСообщения.Информация);

      А_Серверные.ЗаписатьСобытие(ПараметрыСеанса.ТекущийПользователь,
"Ошибка преобразования из Unicode");
      КонецПопытки;

      Возврат Результат;
КонецФункции

Функция ФорматированиеСтроки(Значение) Экспорт

      //Значение = Прав(Значение,СтрДлина(Значение)-1);
      Значение = UnicodeEncode(Значение);
      Значение = СтрЗаменить(Значение, "\n",Символы.ПС);
      Возврат Значение;

КонецФункции

Процедура GptComment(ЗаявкаССылка, Мастер) Экспорт

      //А_Серверные.ЗаписатьСобытие(ЗаявкаССылка, "GptComment");

      ИмяФункцииДляИсполнения=("А_Питон.GptComment");
      ВыполнятьСтандартно    = Истина;

      Отказ = Ложь;

      Попытка


      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПередПроцедурой,""));

            Если ВыполнятьСтандартно Тогда

                  Если ЗначениеЗаполнено(ЗаявкаССылка) И
ЗначениеЗаполнено(Мастер) И НЕ
Мастер.АвтоматическийАссистентКомментариевЗаявкиОтключен Тогда

                        api_key = Константы.openai_key.Получить();
```

```
                        Если НЕ ЗначениеЗаполнено(api_key) Тогда
                              //A_Серверные.ЗаписатьСобытие(ЗаявкаССылка,
"Отказ [api_key]: "+api_key);
                              Отказ = Истина;
                        КонецЕсли;

                        Если НЕ Отказ Тогда
                              // Проверим, есть ли в комментариях знак
вопроса
                              Запрос = Новый запрос;
                              Запрос.УстановитьПараметр("Заявка",
ЗаявкаССылка);
                              Запрос.УстановитьПараметр("Пользователь",
Справочники.Пользователи.НайтиПоНаименованию("mrm"));
                              Запрос.Текст = "ВЫБРАТЬ
                                          | A_ЗаявкаКомментарии.Заявка
КАК КоличествоКомментариев
                                          |ИЗ
                                          |
      РегистрСведений.A_ЗаявкаКомментарии КАК A_ЗаявкаКомментарии
                                          |ГДЕ
                                          | A_ЗаявкаКомментарии.Заявка =
&Заявка
                                          | И
A_ЗаявкаКомментарии.Комментарий ПОДОБНО ""%?%""
                                          | И
A_ЗаявкаКомментарии.Пользователь = &Пользователь
                                          |
                                          |СГРУППИРОВАТЬ ПО
                                          | A_ЗаявкаКомментарии.Заявка";
                              Выборка = Запрос.Выполнить().Выбрать();
                              Если НЕ Выборка.Количество() Тогда
                                    // Если знака вопроса от мастера нет,
то не учавствуем в переписке
                                    Отказ = Истина;
                              КонецЕсли;
                        КонецЕсли;

                        // Информация о заявке
                        ИнформацияЗаявки = "Вы ассистент мастера по
заявке, в компании занимающейся ремонтом бытовой техники.
                        //|User это мастер. Если мастер просто информирует
вас, отвечайте что информация принята. Если спрашивает, окажите ему
поддержку.
                        //|Если мастер просит продлить заявку, ответьте
что запрос выполнен и добавьте в ответ команду [продление заявки]
                        //|Если мастер просит урегулировать конфликт с
клиентом, ответьте что задача для отк создана и добавьте в ответ команду
[внимание отк]
                        //|Помогайте мастеру получить наибольший доход и
выполнить работу наиболее качественно.";
                        |Помогайте мастеру получить информацию которая его
интересует, если вопрос не риторический.
                        |Не указывайте ваше имя в ваших сообщениях.";

                        Если НЕ Отказ Тогда

                              Если ЗначениеЗаполнено(Мастер.ПарольАК) Тогда
```

```
                                    ИнформацияЗаявки = ИнформацияЗаявки +
Символы.ПС + "Код мастера: "+Мастер.ПарольАК;
                                    ИнформацияЗаявки = ИнформацияЗаявки +
Символы.ПС + "Пароль авктокоммутатора (АК): "+Мастер.ПарольАК;
                                    ИнформацияЗаявки = ИнформацияЗаявки +
Символы.ПС + "Кодовое слово: "+Мастер.ПарольАК;
                              КонецЕсли;

                              Запрос = Новый Запрос;
                              Запрос.УстановитьПараметр("Заявка",
ЗаявкаССылка);
                              Запрос.Текст = "ВЫБРАТЬ
                                          | А_Заявка.Номер,
                                          | А_Заявка.ПредставлениеАдреса
КАК Адрес,
                                          |
      А_Заявка.Направление.Примечание КАК Направление,
                                          | А_Заявка.Бренд,
                                          | А_Заявка.ТоварнаяГруппа,
                                          |
      А_Заявка.ЗаявленнаяНеисправность1 как ЗаявленнаяНеисправность,
                                          | А_Заявка.Модель,
                                          |
      А_Заявка.СерийныйНомерАппарата,
                                          |
      А_Заявка.ПродуктовыйНомерАппарата,
                                          |
      А_Заявка.ПримечаниеКНеисправности,
                                          |
      А_Заявка.ПримечаниеДляКлиента,
                                          | А_Заявка.Примечание
                                          |ИЗ
                                          | Документ.А_Заявка КАК
А_Заявка
                                          |ГДЕ
                                          | А_Заявка.Ссылка = &Заявка";
                              Выборка = Запрос.Выполнить().Выбрать();
                              Если Выборка.Следующий() Тогда
                                    Если ЗначениеЗаполнено(Выборка.Номер)
Тогда
                                          ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Номер заявки: " + Выборка.Номер;
                                    КонецЕсли;
                                    Если ЗначениеЗаполнено(Выборка.Адрес)
Тогда
                                          ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Адрес: " + Выборка.Адрес;
                                    КонецЕсли;
                                    Если
ЗначениеЗаполнено(Выборка.Направление) Тогда
                                          ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Направление: " + Выборка.Направление;
                                    КонецЕсли;
                                    Если
ЗначениеЗаполнено(Выборка.ТоварнаяГруппа) Тогда
                                          ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Товарная группа: " +
Выборка.ТоварнаяГруппа;
                                    КонецЕсли;
```

```
                                    Если
ЗначениеЗаполнено(Выборка.ЗаявленнаяНеисправность) Тогда
                                        ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Заявленная неисправность: " +
Выборка.ЗаявленнаяНеисправность;
                                    КонецЕсли;
                                    Если
ЗначениеЗаполнено(Выборка.СерийныйНомерАппарата) Тогда
                                        ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Серийный номер техники: " +
Выборка.СерийныйНомерАппарата;
                                    КонецЕсли;
                                    Если
ЗначениеЗаполнено(Выборка.ПродуктовыйНомерАппарата) Тогда
                                        ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Продуктовый номер техники: " +
Выборка.ПродуктовыйНомерАппарата;
                                    КонецЕсли;
                                    Если
ЗначениеЗаполнено(Выборка.ПримечаниеКНеисправности) Тогда
                                        ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Примечание к неисправности: " +
Выборка.ПримечаниеКНеисправности;
                                    КонецЕсли;
                                    Если
ЗначениеЗаполнено(Выборка.ПримечаниеДляКлиента) Тогда
                                        ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Примечание для клиента: " +
Выборка.ПримечаниеДляКлиента;
                                    КонецЕсли;
                                    Если
ЗначениеЗаполнено(Выборка.Примечание) Тогда
                                        ИнформацияЗаявки =
ИнформацияЗаявки + Символы.ПС + "Примечание: " + Выборка.Примечание;
                                    КонецЕсли;
                                КонецЕсли;

                                //Диалог с клиентом
                                //    select distinct top 100
                                //    stt.audio_file_name,
                                //    stt.record_date,
                                //    stt.start,
                                //    stt.conf,
                                //    stt.sentiment_pos,
                                // stt.sentiment_neg,
                                //    stt.side,
                                //    stt.source_id,
                                //    stt.src,
                                //    stt.dst,
                                //    stt.text,
                                //    stt.linkedid,
                                // calls.oper,
                                // calls.oper_name,
                                // calls.base_name,
                                // calls.bid_id,
                                // calls.call_id,
                                // calls.ad
                                //from transcribations as stt
                                //left join calls as calls
```

```
                              // on stt.linkedid=calls.linkedid where
stt.text<>'' and stt.linkedid = '1643172326.3799777' order by stt.start;

                              // Заполнение первой части запроса к OpenAI
                              user_prompt = "[
                                    | {""role"": ""system"", ""content"":
"""+ИнформацияЗаявки+"""}";

                              // Список уже существующих комментариев
заявки
                              Запрос = Новый Запрос;
                              Запрос.УстановитьПараметр("Заявка",
ЗаявкаССылка);
                              Запрос.Текст = "ВЫБРАТЬ
                                          | А_ЗаявкаКомментарии.Период
КАК Период,
                                          | А_ЗаявкаКомментарии.Мастер,
                                          |
      А_ЗаявкаКомментарии.Пользователь,
                                          |
      А_ЗаявкаКомментарии.Комментарий,
                                          |
      А_ЗаявкаКомментарии.Отправитель
                                          |ИЗ
                                          |
      РегистрСведений.А_ЗаявкаКомментарии КАК А_ЗаявкаКомментарии
                                          |ГДЕ
                                          | А_ЗаявкаКомментарии.Заявка =
&Заявка
                                          |
                                          |УПОРЯДОЧИТЬ ПО
                                          | Период";
                              Выборка = Запрос.Выполнить().Выбрать();
                              Если Выборка.Количество() Тогда

                                    Пока Выборка.Следующий() Цикл
                                          Если Выборка.Отправитель =
"ChatGPT" Тогда
                                                user_prompt = user_prompt +
" ,{""role"": ""assistant"", ""content"": ""Ассистент:
"+Выборка.Комментарий+"""} ";
                                          ИначеЕсли
ЗначениеЗаполнено(Выборка.Мастер) Тогда
                                                user_prompt = user_prompt +
" ,{""role"": ""user"", ""content"": ""Мастер: "+Выборка.Комментарий+"""}
";
                                          Иначе
                                                user_prompt = user_prompt +
" ,{""role"": ""user"", ""content"": """+Выборка.Отправитель+":
"+Выборка.Комментарий+"""} ";
                                          КонецЕсли;
                                    КонецЦикла;
                              Иначе
                                    Отказ = Истина;

      //А_Серверные.ЗаписатьСобытие(ЗаявкаССылка, "Отказ [Список уже
существующих комментариев заявки]: "+Выборка.Количество());
                              КонецЕсли;

                              // Заполнение второй части запроса к OpenAI
```

```
                              user_prompt = user_prompt +" ]";
                              prompt = "{
                                 | ""api_key"": """+api_key+""",
                                 | ""model"": ""gpt-3.5-turbo"",
                                 | ""prompt"": "+user_prompt+"
                                 |}";

                        КонецЕсли;



                        Если НЕ Отказ Тогда

                              prompt = СтрЗаменить(prompt, Символы.ПС, "
");

                              //A_Серверные.ЗаписатьСобытие(ЗаявкаССылка,
"GptComment. Ok [0]: "+prompt);

                              Сервер                = "10.2.4.164";
                              Порт            = 4714;
                              ПутьНаСервере    = "/request";

                              Соединение = Новый HTTPСоединение(Сервер,
Порт);
                              ЗапросСервера = Новый
HTTPЗапрос(ПутьНаСервере);
                              ЗапросСервера.УстановитьТелоИзСтроки(Prompt);
                              ОтветСервера =
Соединение.ВызватьHTTPМетод("POST", ЗапросСервера); //8.3
                              //ОтветСервера =
Соединение.ОтправитьДляОбработки(ЗапросСервера); //8.2
                              КодОтвета = ОтветСервера.КодСостояния;
                              Response =
ОтветСервера.ПолучитьТелоКакСтроку();
                              //A_Серверные.ЗаписатьСобытие(ЗаявкаССылка,
"GptComment. Ok [1]: "+Response);

                              Ответ =
A_РаботаCJSON.ЗаполнитьСтруктуруИзОтветаJSON836(Response);
                              ПоследнийЭлемент =
Ответ.choices.Количество()-1;
                              ТекстОтвета =
ФорматированиеСтроки(Ответ.choices[ПоследнийЭлемент].message.content);
                              //A_Серверные.ЗаписатьСобытие(ЗаявкаССылка,
"GptComment. Ok [2]: "+ТекстОтвета);

                              // Добавить сообщение с уведомлением мастеру
                              НаборЗаписей =
РегистрыСведений.A_ЗаявкаКомментарии.СоздатьНаборЗаписей();

                              ТекПериод = ТекущаяДата();


      НаборЗаписей.Отбор.Заявка.Установить(ЗаявкаССылка);

      НаборЗаписей.Отбор.Мастер.Установить(Справочники.A_Мастера.ПустаяСс
ылка());

      НаборЗаписей.Отбор.Период.Установить(ТекПериод);
```

```
                              НоваяЗапись = НаборЗаписей.Добавить();

                              НоваяЗапись.Заявка          = ЗаявкаСсылка;
                              НоваяЗапись.Мастер          =
Справочники.А_Мастера.ПустаяСсылка();
                              НоваяЗапись.Период          = ТекПериод;
                              НоваяЗапись.Пользователь=
ПараметрыСеанса.ТекущийПользователь;
                              НоваяЗапись.Отправитель     = "ChatGPT";
                              НоваяЗапись.Комментарий = ТекстОтвета;
                              НоваяЗапись.GptTokensTotal =
Число(Ответ.usage.total_tokens);
                              //А_Серверные.ЗаписатьСобытие(ЗаявкаСсылка,
"GptComment. Ok [3]: "+Ответ.usage.total_tokens);
                              НоваяЗапись.Будильник  = Истина;

                              НаборЗаписей.Записать(Истина);


      МРМ_ВебСервис.ОтправитьПушПоКомментарию(Справочники.А_Мастера.Пуста
яСсылка(),ЗаявкаСсылка,ТекстОтвета,0,НоваяЗапись.Пользователь,Истина);

                        Иначе
                              //А_Серверные.ЗаписатьСобытие(ЗаявкаСсылка,
"Отказ [prompt]: "+prompt);
                              ;
                        КонецЕсли;

                  Иначе
                        А_Серверные.ЗаписатьСобытие(ЗаявкаСсылка,
"GptComment. No condition: "+ЗначениеЗаполнено(ЗаявкаСсылка)+"
"+ЗначениеЗаполнено(Мастер)+"
"+Мастер.АвтоматическийАссистентКомментариевЗаявки);
                        ;
                  КонецЕсли;

            КонецЕсли;


      Выполнить(?(ЗначениеЗаполнено(Справочники.А_ВебСервисКоды.НайтиПоНа
именованию(ИмяФункцииДляИсполнения,Истина)),Справочники.А_ВебСервисКоды.Н
айтиПоНаименованию(ИмяФункцииДляИсполнения,Истина).ПослеПроцедуры,""));

      Исключение

            А_Серверные.ЗаписатьСобытие(ЗаявкаСсылка, "GptComment.
"+ОписаниеОшибки());

      КонецПопытки;

КонецПроцедуры
```