CriteriaLoader – класс по формированию критериев для загрузки со скриптом

- loaders[0] {NHibernate.Loader.Criteria.CriteriaLoader(SELECT this\_.CustomerId as CustomerId0\_0\_, this\_.FirstName as FirstName0\_0\_, this\_.LastName as LastName0\_0\_ FROM Customer this\_)} NHibernate.Loader.Criteria.CriteriaLoader

+ ((NHibernate.Loader.OuterJoinLoader)(loaders[0])).sql

{SELECT this\_.CustomerId as CustomerId0\_0\_, this\_.FirstName as FirstName0\_0\_, this\_.LastName as LastName0\_0\_ FROM Customer this\_}

NHibernate.SqlCommand.SqlString

Вопрос по классу Nhibernate.ArrayHelper

Базовая цепочка событий

+ results Count = 0 System.Collections.IList {System.Collections.Generic.List<ItCuties.NHibernate.Customer>}

Важный вопрос 0 tx.CommitTransaction

FlushEvent – сохранение данных

flushEventListener[i].OnFlush(new FlushEvent(this));

IDbTransaction : IDisposable

var tx = session.BeginTransaction() – AdoTransaction

Получение данных Session.List

Класс SessionIml List

Анализ логики чтения данных.

1)Создание конфигурации

var cfg = new Configuration(); - создание объекта конфигурации.

cfg.DataBaseIntegration – вызов метода по настройке интеграции с БД.

ConfigurationExtensions

Делегат – объект, который определяется сигнатурой метода и возвращаемым типом.

Далее к нему могут подключаться ссылки на методы,

которым он будет делегировать выполнение действий с объектами, которые

происходят при вызове сигнатуры базового определенного для него метода.

Получение данных:

Главный метод – List

Интересный критерий отбора с формированием public ICriteria CreateCriteria<T>() where T : class.

Реализация – CriteriaImpl

Формирование данных - session.List(this, results); CriteriaImpl.

Далее при реализации вызывется

string[] implementors = Factory.GetImplementors(criteria.EntityOrClassName);

implementors –

Проблема string[] implementors – пустой массив.

string[] implementors = Factory.GetImplementors(criteria.EntityOrClassName);

SessionImpl.cs

Для проверки данной ошибки был развернут рабочий пример работы с Nh.

Далее в нем был заменена ссылка на библиотеку с исходным кодом Nh.

После этого был включен Debug рабочей версии.

Проблема в получении классов, которые будут реализовывать требуемый функционал:

implementors = Factory.GetImplementors(criteria.EntityOrClassName);

Для рабочего варианта сработало условие SessionFactoryImpl

// NH Different implementation for performance: a class without at least a namespace sure can't be found by reflection

if (entityOrClassName.IndexOf('.') > 0)

{

IEntityPersister checkPersister;

// NH Different implementation: we have better performance checking, first of all, if we know the class

// and take the System.Type directly from the persister (className have high probability to be entityName at least using Criteria or Linq)

if (entityPersisters.TryGetValue(entityOrClassName, out checkPersister))

{

if (!checkPersister.EntityMetamodel.HasPocoRepresentation)

{

// we found the persister but it is a dynamic entity without class

knownMap = new[] { entityOrClassName };

entityNameImplementorsMap[entityOrClassName] = knownMap;

return knownMap;

}

// NH : take care with this because we are forcing the Poco EntityMode

clazz = checkPersister.GetMappedClass(EntityMode.Poco);

}

Далее самый важный объект entityPersisters в SessionFactoryImpl.

var results = new List<string>();

foreach (var q in entityPersisters.Values.OfType<IQueryable>())

При вызове if (entityPersisters.TryGetValue(entityName, out value) == false)

+ [NHibernate.Persister.Entity.SingleTableEntityPersister] {SingleTableEntityPersister(ItCuties.NHibernate.Customer)} NHibernate.Persister.Entity.SingleTableEntityPersister

На основании «ItCuties.NHibernate.Customer»

Формируются загрузчики:

for (int i = 0; i < size; i++)

{

loaders[i] = new CriteriaLoader(

GetOuterJoinLoadable(implementors[i]),

Factory,

criteria,

implementors[i],

enabledFilters

);

spaces.UnionWith(loaders[i].QuerySpaces);

}

CriteriaLoader –

Создается translator = new CriteriaQueryTranslator(factory, rootCriteria, rootEntityName, CriteriaQueryTranslator.RootSqlAlias);

querySpaces = translator.GetQuerySpaces(); - название таблицы

querySpaces[0] = Customer

Добавление данных ArrayHelper.AddAll(results, loaders[i].List(this)); SessionImpl.cs

Далее

public IList List(ISessionImplementor session)

{

return List(session, translator.GetQueryParameters(), querySpaces, resultTypes);

}

private IList ListIgnoreQueryCache(ISessionImplementor session, QueryParameters queryParameters)

{

return GetResultList(DoList(session, queryParameters), queryParameters.ResultTransformer);

}

Loader.cs

result = DoQueryAndInitializeNonLazyCollections(session, queryParameters, true, forcedResultTransformer);

* private IList DoQueryAndInitializeNonLazyCollections(ISessionImplementor session, QueryParameters queryParameters, bool returnProxies, IResultTransformer forcedResultTransformer)

Loader.cs

persistenceContext.BeforeLoad();

Далее в методе вызов

result = DoQuery(session, queryParameters, returnProxies, forcedResultTransformer);

private IList DoQuery(ISessionImplementor session, QueryParameters queryParameters, bool returnProxies, IResultTransformer forcedResultTransformer)

IDbCommand st = PrepareQueryCommand(queryParameters, false, session);

Реализуется интерфейс IDbCommand.

IDbCommand : IDisposable

queryParameters

+ queryParameters {NHibernate.Engine.QueryParameters} NHibernate.Engine.QueryParameters

+ session {NHibernate.Impl.SessionImpl} NHibernate.Engine.ISessionImplementor {NHibernate.Impl.SessionImpl}

Вызов:

ISqlCommand sqlCommand = CreateSqlCommand(queryParameters, session);

Дополнительная информация:

/// <summary>

/// Obtain an <c>IDbCommand</c> with all parameters pre-bound. Bind positional parameters,

/// named parameters, and limit parameters.

/// </summary>

/// <remarks>

/// Creates an IDbCommand object and populates it with the values necessary to execute it against the

/// database to Load an Entity.

/// </remarks>

/// <param name="queryParameters">The <see cref="QueryParameters"/> to use for the IDbCommand.</param>

/// <param name="scroll">TODO: find out where this is used...</param>

/// <param name="session">The SessionImpl this Command is being prepared in.</param>

/// <returns>A CommandWrapper wrapping an IDbCommand that is ready to be executed.</returns>

protected internal virtual IDbCommand PrepareQueryCommand(QueryParameters queryParameters, bool scroll, ISessionImplementor session)

Непосредственная генерация запроса

ISqlCommand sqlCommand = CreateSqlCommand(queryParameters, session); Loader.cs

var parameterSpecs = new HashSet<IParameterSpecification>(GetParameterSpecifications());

SqlString sqlString = SqlString.Copy();

> NHibernate.dll!NHibernate.Driver.SqlClientDriver.CreateCommand() Line 44 C#

> NHibernate.dll!NHibernate.Driver.DriverBase.GenerateCommand(System.Data.CommandType type, NHibernate.SqlCommand.SqlString sqlString, NHibernate.SqlTypes.SqlType[] parameterTypes) Line 131 C#

SetCommandParameters(cmd, parameterTypes);

/////

protected internal virtual IDbCommand PrepareQueryCommand(QueryParameters queryParameters, bool scroll, ISessionImplementor session)

sqlCommand.Bind(command, session);

> NHibernate.dll!NHibernate.Loader.Loader.DoQuery(NHibernate.Engine.ISessionImplementor session, NHibernate.Engine.QueryParameters queryParameters, bool returnProxies, NHibernate.Transform.IResultTransformer forcedResultTransformer) Line 436 C#

IDataReader rs = GetResultSet(st, queryParameters.HasAutoDiscoverScalarTypes, queryParameters.Callable, selection,

session);

IDataReader : IDisposable, IDataRecord - .Net

// TODO NH: Callable

rs = session.Batcher.ExecuteReader(st);

reader = cmd.ExecuteReader();

> NHibernate.dll!NHibernate.AdoNet.AbstractBatcher.ExecuteReader(System.Data.IDbCommand cmd) Line 222 C#

reader = new NHybridDataReader(reader);

> NHibernate.dll!NHibernate.AdoNet.AbstractBatcher.ExecuteReader(System.Data.IDbCommand cmd) Line 241 C#

-----------------------

public override void List(CriteriaImpl criteria, IList results)

{

using (new SessionIdLoggingContext(SessionId))

{

CheckAndUpdateSessionStatus();

string[] implementors = Factory.GetImplementors(criteria.EntityOrClassName);

int size = implementors.Length;

CriteriaLoader[] loaders = new CriteriaLoader[size];

ISet<string> spaces = new HashSet<string>();

for (int i = 0; i < size; i++)

{

loaders[i] = new CriteriaLoader(

GetOuterJoinLoadable(implementors[i]),

Factory,

criteria,

implementors[i],

enabledFilters

);

spaces.UnionWith(loaders[i].QuerySpaces);

}

AutoFlushIfRequired(spaces);

dontFlushFromFind++;

bool success = false;

try

{

for (int i = size - 1; i >= 0; i--)

{

ArrayHelper.AddAll(results, loaders[i].List(this));

}

success = true;

}

catch (HibernateException)

{

// Do not call Convert on HibernateExceptions

throw;

}

catch (Exception sqle)

{

throw Convert(sqle, "Unable to perform find");

}

finally

{

dontFlushFromFind--;

AfterOperation(success);

}

}

}