using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

using GTA.DC.Common.DBUtility;

using GTA.DC.Common.Utilities;

using GTA.DC.MetaFactory.IDAL;

using GTA.DC.MetaFactory.Entity;

using System.Data.OracleClient;

using Microsoft.Practices.EnterpriseLibrary.Data;

using Microsoft.Practices.EnterpriseLibrary.Data.Oracle;

using System.Data.Common;

using System.Transactions;

namespace GTA.DC.MetaFactory.Oracle.DAL

{

public class AuditInformationDAL : IAuditInformationDAL

{

#region 查阅审核操作的详细信息

/// <summary>

/// 查阅审核操作的详细信息

/// </summary>

/// <param name="\_tbid"></param>

/// <returns></returns>

public DataSet QueryTableCheckMessage(string \_tbid)

{

DataSet \_ds = new DataSet();

string connectionString = ConnectionStringManager.ETLDesign;

#region 获取审核主表信息

string cmd\_TableMessage = @" select tab.\* from tbl\_meta\_de\_tablemaster tab where tab.tbid= '" + \_tbid.ToString()+"'";

\_ds = OracleDbHelper.LoadDataSetQuery(connectionString, CommandType.Text, cmd\_TableMessage);

\_ds.Tables[0].TableName = "TableMeta";

#endregion

#region 获取表变更信息

cmd\_TableMessage = @" select docid,businesstype,dbid,

(case UPDATETYPE when 0 then '新增' when 1 then '修改' when 2 then '删除' else '提升' end) UPDATETYPE,

(case UPDATELEVEL when 0 then '文档' when 1 then '主题' when 2 then '表'

when 3 then '字段' when 4 then '调度' else 'ETL流' end

) UPDATELEVEL,

fieldid,fieldphsicalname,fieldname,tbphyiscname,tbid,tablestate,fieldmappingid,

description,beforevalue,aftervalue,

attribute,updateuserid,updateusername ,updatetime from tbl\_meta\_de\_UpdateDetail where tbid='" + \_tbid.ToString() + "' order by updatetime desc ";

DataSet ds1 = new DataSet();

ds1 = OracleDbHelper.LoadDataSetQuery(connectionString,CommandType.Text,cmd\_TableMessage);

ds1.Tables[0].TableName = "ChangeTable";

\_ds.Tables.Add(ds1.Tables[0].Copy());

ds1 = null;

#endregion

return \_ds;

}

#endregion

/// <summary>

/// 查询表元信息

/// </summary>

/// <param name="\_tbid"></param>

/// <returns></returns>

public DataSet GetTableInfo(List<string> \_tbid)

{

DataSet \_dsTable = new DataSet();

string connectionString = ConnectionStringManager.ETLDesign;

string strtbids = "'" + string.Join("','", \_tbid) + "'";

StringBuilder sql = new StringBuilder();

sql.Append("select ordernum,csmartype,phyiscname,chiname,fieldtypename,unitid,engname,ispk,islimit,derivefrom,isnull,");

sql.Append("interpret,languagetype,codingid,chidesc,engdesc,bupdatestate,bstatedesc,actiontype,remark,fldid,tbid");

sql.Append(" from tbl\_meta\_de\_fieldmaster where nvl(isdelete,0) <> 1 and tbid in ({0}) order by ordernum");

string strSql = string.Format(sql.ToString(), strtbids);

\_dsTable = OracleDbHelper.LoadDataSetQuery(connectionString, CommandType.Text, strSql);

return \_dsTable;

}

#region 项目经理审核管理界面

#region 项目经理返回草稿

/// <summary>

/// 项目经理返回草稿

/// </summary>

/// <param name="listTable"></param>

/// <param name="lstAuditDetail"></param>

/// <returns></returns>

public int BackToDraft(List<TblMetaDeTableMasterEntity> listTable, List<TblMetaDeAuditDetailEntity> lstAuditDetail)

{

int result = 0;

string connectionString = ConnectionStringManager.ETLDesign;

//打开数据库连接

OracleDatabase db = new OracleDatabase(connectionString);

DbConnection conn = db.CreateConnection();

conn.Open();

//开启事务

DbTransaction trans = conn.BeginTransaction();

try

{

//更改表状态

if (listTable != null && listTable.Count > 0)

{

ChangeTableStateToDraft(listTable, conn, trans);

}

//插入表审核记录

if (lstAuditDetail != null && lstAuditDetail.Count > 0)

{

InsertAuditDetail(lstAuditDetail, conn, trans);

}

trans.Commit();

result = 1;

}

catch (Exception ex)

{

//保存失败，执行回滚

trans.Rollback();

result = 0;

}

finally

{

conn.Close();

conn.Dispose();

}

return result;

}

/// <summary>

/// 更改表状态

/// </summary>

/// <param name="listTable"></param>

/// <param name="conn"></param>

/// <param name="trans"></param>

/// <returns></returns>

private bool ChangeTableStateToDraft(List<TblMetaDeTableMasterEntity> listTable, DbConnection conn, DbTransaction trans)

{

bool result = true;

string sql = string.Empty;

try

{

sql = @"update tbl\_meta\_de\_tablemaster

set State = :State,

prestate = :prestate,

UpdateUserId = :UpdateUserId,

UpdateDate = sysdate,

Description = :Description

where tbid = :tbid";

DbCommand cmd = conn.CreateCommand();

cmd.Transaction = trans;

cmd.CommandText = sql;

OracleParameter[] parameter = new OracleParameter[]{

new OracleParameter(":State",OracleType.Number),

new OracleParameter(":prestate",OracleType.Number),

new OracleParameter(":tbid",OracleType.VarChar),

new OracleParameter(":UpdateUserId",OracleType.VarChar),

new OracleParameter(":Description",OracleType.VarChar)

};

foreach (TblMetaDeTableMasterEntity entity in listTable)

{

cmd.Parameters.Clear();

parameter[0].Value = entity.State;

parameter[1].Value = entity.PreState;

parameter[2].Value = entity.TBID;

parameter[3].Value = entity.UpdateUserId;

parameter[4].Value = entity.Description;

cmd.Parameters.AddRange(parameter);

cmd.ExecuteNonQuery();

}

result = true;

}

catch (Exception ex)

{

result = false;

throw ex;

}

return result;

}

#endregion

#region 项目经理提交运维

/// <summary>

/// 项目经理提交运维

/// </summary>

/// <param name="listTable"></param>

/// <param name="lstAuditDetail"></param>

/// <returns></returns>

public int SubmitMaintenance(List<TblMetaDeTableMasterEntity> listTable, List<TblMetaDeAuditDetailEntity> lstAuditDetail)

{

int result = 0;

string connectionString = ConnectionStringManager.ETLDesign;

//打开数据库连接

OracleDatabase db = new OracleDatabase(connectionString);

DbConnection conn = db.CreateConnection();

conn.Open();

//开启事务

DbTransaction trans = conn.BeginTransaction();

try

{

//更改表状态

if (listTable != null && listTable.Count > 0)

{

ChangeTableStateToMaintenance(listTable, conn, trans);

}

//插入表审核记录

if (lstAuditDetail != null && lstAuditDetail.Count > 0)

{

InsertAuditDetail(lstAuditDetail, conn, trans);

}

trans.Commit();

result = 1;

}

catch (Exception ex)

{

//保存失败，执行回滚

trans.Rollback();

result = 0;

}

finally

{

conn.Close();

conn.Dispose();

}

return result;

}

/// <summary>

/// 更改表状态

/// </summary>

/// <param name="listTable"></param>

/// <param name="conn"></param>

/// <param name="trans"></param>

/// <returns></returns>

private bool ChangeTableStateToMaintenance(List<TblMetaDeTableMasterEntity> listTable, DbConnection conn, DbTransaction trans)

{

bool result = true;

string sql = string.Empty;

try

{

sql = @"update tbl\_meta\_de\_tablemaster

set State = :State,

prestate = :prestate,

UpdateUserId = :UpdateUserId,

UpdateDate = SYSDATE

where tbid = :tbid";

DbCommand cmd = conn.CreateCommand();

cmd.Transaction = trans;

cmd.CommandText = sql;

OracleParameter[] parameter = new OracleParameter[]{

new OracleParameter(":State",OracleType.Number),

new OracleParameter(":prestate",OracleType.Number),

new OracleParameter(":tbid",OracleType.VarChar),

new OracleParameter(":UpdateUserId",OracleType.VarChar),

};

foreach (TblMetaDeTableMasterEntity entity in listTable)

{

cmd.Parameters.Clear();

parameter[0].Value = entity.State;

parameter[1].Value = entity.PreState;

parameter[2].Value = entity.TBID;

parameter[3].Value = entity.UpdateUserId;

cmd.Parameters.AddRange(parameter);

cmd.ExecuteNonQuery();

}

result = true;

}

catch (Exception ex)

{

result = false;

throw ex;

}

return result;

}

#endregion

#region 插入表审核记录

/// <summary>

/// 插入表审核记录

/// </summary>

/// <param name="lstAuditDetail"></param>

/// <param name="conn"></param>

/// <param name="trans"></param>

/// <returns></returns>

private bool InsertAuditDetail(List<TblMetaDeAuditDetailEntity> lstAuditDetail, DbConnection conn, DbTransaction trans)

{

bool result = true;

string sql = string.Empty;

OracleParameter[] parameterup = new OracleParameter[]

{

new OracleParameter(":TbID",OracleType.VarChar),

new OracleParameter(":OPERATIONSTEP",OracleType.Number),

new OracleParameter(":State",OracleType.Number),

new OracleParameter(":OperUserID",OracleType.VarChar),

new OracleParameter(":OperUserName",OracleType.VarChar),

new OracleParameter(":AuditContent",OracleType.VarChar),

};

try

{

sql = @"insert into tbl\_meta\_de\_AuditDetail(TbID,OPERATIONSTEP,State,OperUserID,OperUserName,OperTime,AuditContent)

values (:TbID,:OPERATIONSTEP,:State,:OperUserID,:OperUserName,SYSDATE,:AuditContent)";

DbCommand cmd = conn.CreateCommand();

cmd.Transaction = trans;

cmd.CommandText = sql.ToString();

foreach (TblMetaDeAuditDetailEntity entity in lstAuditDetail)

{

cmd.Parameters.Clear();

parameterup[0].Value = entity.TbID;

parameterup[1].Value = entity.OperationStep;

parameterup[2].Value = entity.State;

parameterup[3].Value = entity.OperUserID;

parameterup[4].Value = entity.OperUserName;

parameterup[5].Value = entity.AuditContent;

cmd.Parameters.AddRange(parameterup);

cmd.ExecuteNonQuery();

}

result = true;

}

catch (Exception ex)

{

result = false;

throw ex;

}

return result;

}

#endregion

#endregion

#region 项目经理查询审核表信息

public PagingResult ProjMgrQueryTable(string querytext, int tbstate, int pageIndex, int pageSize)

{

PagingResult pagingResult = new PagingResult();

StringBuilder sql = new StringBuilder();

StringBuilder sqlconnect = new StringBuilder();

StringBuilder sqlWhere = new StringBuilder();

List<OracleParameter> querySQLParameterList = new List<OracleParameter>();

sql.Append(@" select tb.docid,doc.docname,buss.business,db.dbid,db.name,tb.tbid,

tb.phyiscname,tb.chiname,

case tb.operationtype when 0 then '新增' when 1 then '修改' when 2 then '删除' when 3 then '提升' end as operationtype,

tb.state,user1.username as createuser,

to\_char(tb.createdate,'yyyy-mm-dd hh24:mi:ss') as createdate,

user2.username as updateuser,

to\_char(tb.updatedate,'yyyy-mm-dd hh24:mi:ss') as updatedate,

tb.developmentuserid,

user3.username as testuser,tb.description ");

if (!string.IsNullOrEmpty(querytext))

{

sql.Append(@" , case when upper(db.dbid) like :querytext escape '/' then length(db.dbid)

when upper(db.name) like :querytext escape '/' then length(db.name)

when upper(tb.tbid) like :querytext escape '/' then length(tb.tbid)

when upper(tb.phyiscname) like :querytext escape '/' then length(tb.phyiscname)

when upper(tb.chiname) like :querytext escape '/' then length(tb.chiname) end as FieldLen

");

}

sql.Append(@" from tbl\_meta\_de\_tablemaster tb ");

sqlWhere.Append("where 1=1 ");

if (Convert.ToInt32(tbstate) >= 0)

{

sqlWhere.Append(" and tb.state = :tbstate");

querySQLParameterList.Add(new OracleParameter(":tbstate", tbstate));

}

if (!string.IsNullOrEmpty(querytext))

{

sqlWhere.Append(@" and (upper(db.dbid) like :querytext escape '/' or upper(db.name) like :querytext escape '/'

or upper(tb.tbid) like :querytext escape '/' or upper(tb.phyiscname) like :querytext escape '/'

or upper(tb.chiname) like :querytext escape '/')

");

querySQLParameterList.Add(new OracleParameter(":querytext", "%" + querytext.ToUpper() + "%"));

}

sqlconnect.Append("left join tbl\_meta\_de\_databasemaster db on tb.dbid=db.dbid ");

sqlconnect.Append("left join tbl\_meta\_de\_document doc on doc.docid=tb.docid ");

sqlconnect.Append("left join tbl\_meta\_de\_businesstype buss on db.typeid=buss.btypeid ");

sqlconnect.Append("left join tbl\_um\_user user1 on tb.createuserid=user1.userid ");

sqlconnect.Append("left join tbl\_um\_user user2 on tb.updateuserid=user2.userid ");

sqlconnect.Append("left join tbl\_um\_user user3 on tb.testuserid=user3.userid ");

string strOrderBy = "";

if (!string.IsNullOrEmpty(querytext))

{

strOrderBy = " order by FieldLen asc,updatedate desc";

}else

strOrderBy = " order by updatedate desc";

string executeSql = sql.ToString() + sqlconnect.ToString() + sqlWhere.ToString() + strOrderBy;

string pageSql = string.Format(@"select \* from(select t.\*,rownum rn from({0}) t where rownum<={1}) where rn>{2} ", executeSql, pageSize \* pageIndex, pageSize \* (pageIndex - 1));

pagingResult.DataSet = OracleDbHelper.LoadDataSetQuery(ConnectionStringManager.ETLDesign, CommandType.Text, pageSql, querySQLParameterList.ToArray());

string countSql = string.Format(@"select count(1) from tbl\_meta\_de\_tablemaster tb {0}", sqlconnect.ToString() + sqlWhere.ToString());

pagingResult.RowCount = Convert.ToInt32(OracleDbHelper.ExecuteScalar(ConnectionStringManager.ETLDesign, CommandType.Text, countSql, querySQLParameterList.ToArray()));

return pagingResult;

}

#endregion

#region 查询具有开发人员角色的用户

public DataSet GetDevelopMentUser()

{

DataSet ds = new DataSet();

string connectionString = ConnectionStringManager.MetaData.ToString();

StringBuilder sql = new StringBuilder();

sql.Append(@" SELECT us.userid AS USERID,us.username as USERNAME

FROM tbl\_um\_user us

WHERE us.userid IN

(SELECT ru.userid

FROM softsys.tbl\_um\_roleuser ru

WHERE ru.roleid IN

(

SELECT ro.roleid FROM softsys.tbl\_um\_role ro WHERE ro.rolename = 'ETL开发'

)

)

AND us.isdisable=0

ORDER BY USERNAME

");

ds = OracleDbHelper.LoadDataSetQuery(connectionString, CommandType.Text, sql.ToString());

return ds;

}

#endregion

#region 查询具有测试人员角色的用户

public DataSet GetTestUser()

{

DataSet ds = new DataSet();

string connectionString = ConnectionStringManager.MetaData.ToString();

StringBuilder sql = new StringBuilder();

sql.Append(@" SELECT us.userid AS USERID,us.username as USERNAME

FROM softsys.tbl\_um\_user us

WHERE us.userid IN

(SELECT ru.userid

FROM softsys.tbl\_um\_roleuser ru

WHERE ru.roleid IN

(

SELECT ro.roleid FROM softsys.tbl\_um\_role ro WHERE ro.rolename = '测试人员'

)

)

AND us.isdisable=0

ORDER BY USERNAME

");

ds = OracleDbHelper.LoadDataSetQuery(connectionString, CommandType.Text, sql.ToString());

return ds;

}

#endregion

#region 抽取生成元数据

/// <summary>

/// 生成元数据

/// </summary>

/// <param name="\_tbid"></param>

/// <returns></returns>

public bool ProduceMeta(Dictionary<string, string> \_tbids)

{

string connectionString = ConnectionStringManager.ETLDesign;//设计库元数据

OracleDatabase db = new OracleDatabase(connectionString);

DbConnection conn = db.CreateConnection();

conn.Open();

DbTransaction trans = conn.BeginTransaction();

try

{

foreach (var \_tbid in \_tbids)

{

OracleParameter[] param = new OracleParameter[]

{

new OracleParameter(":TBID",OracleType.VarChar)

};

param[0].Value = \_tbid.Key;

#region 同步表元

string strSql = @"MERGE INTO tbl\_meta\_tablemaster t

USING (SELECT Tbtype,

Phyiscname,

Chiname,

Engname,

Startdate,

Enddate,

Chidesc,

Engdesc,

Score,

Bupdatestate,

Dbid,

Bstatedesc,

Ordernum,

Updatedate,

tbid

FROM tbl\_meta\_de\_tablemaster

WHERE tbid = :tbid) S

ON (T.tbid = S.tbid)

WHEN MATCHED THEN

UPDATE

SET T.Tbtype = S.Tbtype,

T.Phyiscname = S.Phyiscname,

T.Chiname = S.Chiname,

T.Engname = S.Engname,

T.Startdate = S.Startdate,

T.Enddate = S.Enddate,

T.Chidesc = S.Chidesc,

T.Engdesc = S.Engdesc,

T.Score = S.Score,

T.Bupdatestate = S.Bupdatestate,

T.Dbid = S.Dbid,

T.Bstatedesc = S.Bstatedesc,

T.Ordernum = S.Ordernum,

T.UPDATETIME = SYSDATE

WHERE t.tbid = :tbid

WHEN NOT MATCHED THEN INSERT(Tbid, Tbtype, Phyiscname, Chiname, Engname, Startdate, Enddate, Chidesc, Engdesc, Score, Bupdatestate, Dbid, Bstatedesc, Ordernum, Updatetime)

VALUES(S.Tbid, S.Tbtype,S.Phyiscname, S.Chiname, S.Engname, S.Startdate, S.Enddate, S.Chidesc, S.Engdesc, S.Score, S.Bupdatestate, S.Dbid, S.Bstatedesc, S.Ordernum, sysdate)

";

#endregion

OracleDbHelper.ExecuteNonQuery(conn, trans, CommandType.Text, strSql, param);

#region 生成字段

strSql = @"MERGE INTO tbl\_meta\_fieldMaster t

USING (SELECT Phyiscname,

Chiname,

Engname,

Ispk,

Derivefrom,

Isnull,

Interpret,

Chidesc,

Engdesc,

Remark,

Bupdatestate,

Tbid,

Fldtypeid,

Unitid,

Languagetype,

Intodesc,

Bstatedesc,

Codingid,

Actiontype,

case when Isdelete=1 then 1 else NULL END Isdelete,

Ordernum,

Islimit,

Fieldtypename,

CreateDATE,

Fldid

FROM tbl\_meta\_de\_FieldMaster

WHERE tbid = :tbid) S

ON (T.Fldid = S.Fldid)

WHEN MATCHED THEN

UPDATE

SET T.Phyiscname = S.Phyiscname,

T.Chiname = S.Chiname,

T.Engname = S.Engname,

T.Ispk = S.Ispk,

T.Derivefrom = S.Derivefrom,

T.Isnull = S.Isnull,

T.Interpret = S.Interpret,

T.Chidesc = S.Chidesc,

T.Engdesc = S.Engdesc,

T.Remark = S.Remark,

T.Bupdatestate = S.Bupdatestate,

T.Tbid = S.Tbid,

T.Fldtypeid = S.Fldtypeid,

T.Unitid = S.Unitid,

T.Languagetype = S.Languagetype,

T.Intodesc = S.Intodesc,

T.Bstatedesc = S.Bstatedesc,

T.Codingid = S.Codingid,

T.Actiontype = S.Actiontype,

T.Isdelete = S.Isdelete,

T.Ordernum = S.Ordernum,

T.Islimit = S.Islimit,

T.Fieldtypename = S.Fieldtypename

WHERE t.tbid = :tbid

WHEN NOT MATCHED THEN

INSERT(Fldid, Phyiscname, Chiname, Engname, Ispk, Derivefrom, Isnull, Interpret, Chidesc, Engdesc, Remark, Bupdatestate, Tbid, Fldtypeid, Unitid,

Languagetype, Intodesc, Bstatedesc, Codingid, Actiontype, Isdelete, Ordernum, Islimit, Fieldtypename, Createtime)

VALUES(S.Fldid, S.Phyiscname,S.Chiname, S.Engname, S.Ispk, S.Derivefrom, S.Isnull, S.Interpret, S.Chidesc, S.Engdesc, S.Remark, S.Bupdatestate, S.Tbid, S.Fldtypeid, S.Unitid,

S.Languagetype, S.Intodesc, S.Bstatedesc, S.Codingid, S.Actiontype, S.Isdelete, S.Ordernum, S.Islimit, S.Fieldtypename, SYSDATE) ";

#endregion

OracleDbHelper.ExecuteNonQuery(conn, trans, CommandType.Text, strSql, param);

#region 生成主题域信息

strSql = @"MERGE INTO tbl\_meta\_databaseMaster t

USING (SELECT Name,

Typeid,

Chidesc,

Engdesc,

Startdate,

Enddate,

UpdateUserid,

Updatetime,

DBID,Type,parentId

from tbl\_meta\_de\_databaseMaster db

CONNECT BY PRIOR parentid = dbid

START WITH dbid in (select dbid

from tbl\_meta\_de\_tableMaster

where tbid = :tbid)) S

ON (T.DBID = S.DBID)

WHEN MATCHED THEN

UPDATE

SET T.Name = S.Name,

T.Typeid = S.Typeid,

T.Chidesc = S.Chidesc,

T.Engdesc = S.Engdesc,

T.Startdate = S.Startdate,

T.Enddate = S.Enddate,

T.Updateserid = S.UpdateUserid,

T.Updatetime = S.Updatetime,

T.Type=S.Type,

t.parentId=s.parentId

WHERE dbid in (SELECT DBID

from tbl\_meta\_databaseMaster db

CONNECT BY PRIOR

parentid = dbid

START WITH dbid IN

(select dbid

from tbl\_meta\_de\_tableMaster

where tbid =

:tbid))

WHEN NOT MATCHED THEN INSERT(Dbid, Name, Typeid, Chidesc, Engdesc, Startdate, Enddate, Updateserid, Updatetime,Type,parentId)

VALUES(S.Dbid, S.Name, S.Typeid, S.Chidesc, S.Engdesc, S.Startdate, S.Enddate, S.UpdateUserid, SYSDATE,S.Type,S.parentId) ";

#endregion

OracleDbHelper.ExecuteNonQuery(conn, trans, CommandType.Text, strSql, param);

}

trans.Commit();

}

catch (Exception ex)

{

trans.Rollback();

return false;

}

finally

{

conn.Close();

conn.Dispose();

}

return true;

}

#endregion

#region 分配开发人员

/// <summary>

/// 分配开发人员

/// </summary>

/// <param name="tableEntitys"></param>

/// <returns></returns>

public bool AssignETLDevelopment(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> lstAuditDetail)

{

bool result = true;

StringBuilder sql = new StringBuilder();

string connectionString = ConnectionStringManager.ETLDesign.ToString();

OracleParameter[] parametertb = new OracleParameter[]

{

new OracleParameter(":State",OracleType.Number),

new OracleParameter(":DevelopmentUserId",OracleType.VarChar),

new OracleParameter(":UpdateUserId",OracleType.VarChar),

new OracleParameter(":OPERATIONSTEP",OracleType.Number),

new OracleParameter(":OPERUSERID",OracleType.VarChar),

new OracleParameter(":OPERUSERNAME",OracleType.VarChar),

new OracleParameter(":AUDITCONTENT",OracleType.VarChar),

new OracleParameter(":tbid",OracleType.VarChar)

};

OracleConnection conn = new OracleConnection(connectionString);

try

{

sql = new StringBuilder();

sql.Append(@"begin update tbl\_meta\_de\_tablemaster set preState=state,state=:State,developmentUserId=:DevelopmentUserId, UpdateUserId =:UpdateUserId,UpdateDate =SYSDATE,");

sql.Append(@" developmentDate=SYSDATE ");

sql.Append(@" where tbid=:tbid;");

sql.Append(@" delete from tbl\_meta\_de\_tablemaster where tbid=:tbid and isDelete=1;");

sql.Append(@" delete from tbl\_meta\_de\_Fieldmaster where tbid=:tbid and isDelete=1;");

sql.Append(@" delete from tbl\_meta\_de\_FieldMapping where tartbid=:tbid and isDelete=1; ");

sql.Append(@" insert into tbl\_meta\_de\_AuditDetail(tbid, operationstep,state, operuserid,operusername,opertime,auditcontent)

values(:tbid,:OPERATIONSTEP,:STATE,:OPERUSERID,:OPERUSERNAME,sysdate,:AUDITCONTENT);

end;");

conn.Open();

using (OracleCommand cmd = conn.CreateCommand())

{

cmd.CommandText = sql.ToString();

foreach (TblMetaDeTableMasterEntity tableEntity in tableEntitys)

{

cmd.Parameters.Clear();

parametertb[0].Value = (int)ETableState.Developing;

parametertb[1].Value = tableEntity.DevelopmentUserId;

//parametertb[2].Value = DateTime.Now;

parametertb[2].Value = tableEntity.UpdateUserId;

//parametertb[4].Value = DateTime.Now;

TblMetaDeAuditDetailEntity auditDetail = lstAuditDetail.Where(t => t.TbID == tableEntity.TBID).FirstOrDefault();

if (auditDetail != null)

{

parametertb[3].Value = auditDetail.OperationStep;

parametertb[4].Value = auditDetail.OperUserID;

parametertb[5].Value = auditDetail.OperUserName;

parametertb[6].Value = auditDetail.AuditContent;

}

else

{

parametertb[3].Value = (int)EAuditDetail.DistributDevelopment;

parametertb[4].Value = "";

parametertb[5].Value = "";

parametertb[6].Value = "";

}

parametertb[7].Value = tableEntity.TBID;

cmd.Parameters.AddRange(parametertb);

cmd.ExecuteNonQuery();

}

}

result = true;

}

catch (Exception ex)

{

result = false;

throw ex;

}

finally

{

if (conn != null)

{

conn.Close();

conn.Dispose();

}

}

return result;

}

#endregion

#region 查询审核表信息

public PagingResult QueryTableAuditInfo(TableQueryParam QueryParam)

{

PagingResult pagingResult = new PagingResult();

StringBuilder sql = new StringBuilder();

StringBuilder sqlconnect = new StringBuilder();

StringBuilder sqlWhere = new StringBuilder();

List<OracleParameter> querySQLParameterList = new List<OracleParameter>();

sql.Append(@" select tb.docid,doc.docname,buss.business,db.dbid,db.name,tb.tbid,

tb.phyiscname,tb.chiname,

case tb.operationtype when 0 then '新增' when 1 then '修改' when 2 then '删除' when 3 then '提升' end as operationtype,

tb.state,user1.username as createuser,

to\_char(tb.createdate,'yyyy-mm-dd hh24:mi:ss') as createdate,

user2.username as updateuser,

to\_char(tb.updatedate,'yyyy-mm-dd hh24:mi:ss') as updatedate,

tb.description,tb.ISCREATEDMETADATA,");

switch(QueryParam.RoleName)

{

case "项目经理":

sql.Append(@"tb.developmentuserid,

user4.username as testuser ");

break;

case "开发人员":

sql.Append(@"user3.username as developmentuser,

tb.testuserid as testuserid ");

break;

default:

sql.Append(@"user3.username as developmentuser,

user4.username as testuser ");

break;

}

if (!string.IsNullOrEmpty(QueryParam.QueryText))

{

sql.Append(@" , case when upper(db.dbid) like :querytext escape '/' then length(db.dbid)

when upper(db.name) like :querytext escape '/' then length(db.name)

when upper(tb.tbid) like :querytext escape '/' then length(tb.tbid)

when upper(tb.phyiscname) like :querytext escape '/' then length(tb.phyiscname)

when upper(tb.chiname) like :querytext escape '/' then length(tb.chiname) end as FieldLen

");

}

sql.Append("from tbl\_meta\_de\_tablemaster tb ");

sqlWhere.Append(" where 1=1 ");

if (Convert.ToInt32(QueryParam.TbState) >= 0)

{

sqlWhere.Append(" and tb.state = :tbstate");

querySQLParameterList.Add(new OracleParameter(":tbstate", QueryParam.TbState));

}

if (!string.IsNullOrEmpty(QueryParam.QueryText))

{

sqlWhere.Append(@" and (upper(db.dbid) like :querytext escape '/' or upper(db.name) like :querytext escape '/'

or upper(tb.tbid) like :querytext escape '/' or upper(tb.phyiscname) like :querytext escape '/'

or upper(tb.chiname) like :querytext escape '/')

");

querySQLParameterList.Add(new OracleParameter(":querytext", "%" + QueryParam.QueryText.ToUpper() + "%"));

}

switch (QueryParam.RoleName)

{

case "开发人员":

sqlWhere.Append(@" and tb.developmenduserid=:userid ");

querySQLParameterList.Add(new OracleParameter(":userid", QueryParam.UserID));

break;

case "测试人员":

sqlWhere.Append(@" and tb.testuserid=:userid ");

querySQLParameterList.Add(new OracleParameter(":userid", QueryParam.UserID));

break;

}

if (!string.IsNullOrEmpty(QueryParam.BusinessType))

{

sqlWhere.Append(@"and tb.dbid in(

select db.dbid

from tbl\_meta\_de\_databasemaster db

start with parentid = :businesstype connect by parentid= prior DBID)

");

querySQLParameterList.Add(new OracleParameter(":businesstype", QueryParam.BusinessType));

}

if (!string.IsNullOrEmpty(QueryParam.DbID))

{

sqlWhere.Append(@"and tb.dbid in(

select db.dbid

from tbl\_meta\_de\_databasemaster db

start with dbid = :dbid connect by prior dbid=parentid)

");

querySQLParameterList.Add(new OracleParameter(":dbid", QueryParam.DbID));

}

if (!string.IsNullOrEmpty(QueryParam.TbID))

{

sqlWhere.Append(" and tb.tbid =:tbid ");

querySQLParameterList.Add(new OracleParameter(":tbid", QueryParam.TbID));

}

sqlconnect.Append("left join tbl\_meta\_de\_document doc on tb.docid=doc.docid ");

sqlconnect.Append("left join tbl\_meta\_de\_databasemaster db on tb.dbid=db.dbid ");

sqlconnect.Append("left join tbl\_meta\_de\_businesstype buss on db.typeid=buss.btypeid ");

sqlconnect.Append("left join tbl\_um\_user user1 on tb.createuserid=user1.userid ");

sqlconnect.Append("left join tbl\_um\_user user2 on tb.updateuserid=user2.userid ");

switch (QueryParam.RoleName)

{

case "项目经理":

sqlconnect.Append("left join tbl\_um\_user user4 on tb.testuserid=user4.userid ");

break;

case "开发人员":

sqlconnect.Append("left join tbl\_um\_user user3 on tb.developmentuserid=user3.userid ");

break;

default:

sqlconnect.Append(@"left join tbl\_um\_user user3 on tb.developmentuserid=user3.userid

left join tbl\_um\_user user4 on tb.testuserid=user4.userid");

break;

}

string strOrderBy = "";

if (!string.IsNullOrEmpty(QueryParam.QueryText))

{

strOrderBy = " order by FieldLen asc,updatedate desc";

}

else

strOrderBy = " order by updatedate desc";

string executeSql = sql.ToString() + sqlconnect.ToString() + sqlWhere.ToString() + strOrderBy;

string pageSql = string.Format(@"select \* from(select t.\*,rownum rn from({0}) t where rownum<={1}) where rn>{2} ", executeSql, QueryParam.PageSize \* QueryParam.PageIndex, QueryParam.PageSize \* (QueryParam.PageIndex - 1));

pagingResult.DataSet = OracleDbHelper.LoadDataSetQuery(ConnectionStringManager.ETLDesign, CommandType.Text, pageSql, querySQLParameterList.ToArray());

string countSql = string.Format(@"select count(1) from tbl\_meta\_de\_tablemaster tb {0}", sqlconnect.ToString() + sqlWhere.ToString());

pagingResult.RowCount = Convert.ToInt32(OracleDbHelper.ExecuteScalar(ConnectionStringManager.ETLDesign, CommandType.Text, countSql, querySQLParameterList.ToArray()));

return pagingResult;

}

#endregion

#region 运维人员操作界面

#region 试运行未通过

public OperationResult TestRunReject(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

OperationResult result = new OperationResult();

StringBuilder sql = new StringBuilder();

string connectionString = ConnectionStringManager.ETLDesign.ToString();

OracleParameter[] parametertb = new OracleParameter[]

{

new OracleParameter(":STATE",OracleType.Number),

new OracleParameter(":TBID",OracleType.VarChar),

new OracleParameter(":OPERATIONSTEP",OracleType.Number),

new OracleParameter(":OPERUSERID",OracleType.VarChar),

new OracleParameter(":OPERUSERNAME",OracleType.VarChar),

new OracleParameter(":AUDITCONTENT",OracleType.VarChar),

//new OracleParameter(":DATE",OracleType.DateTime),

new OracleParameter(":UpdateUserId",OracleType.VarChar)

};

OracleConnection conn = new OracleConnection(connectionString);

OracleCommand cmd = conn.CreateCommand();

try

{

sql = new StringBuilder();

sql.Append(@"begin update tbl\_meta\_de\_tablemaster set preState=state,state=:State,UpdateUserId =:UpdateUserId,UpdateDate =sysdate,description='【运维】：'||:AUDITCONTENT ");

sql.Append(@" where tbid=:tbid;");

sql.Append(@" insert into tbl\_meta\_de\_AuditDetail(tbid, operationstep,state, operuserid,operusername,opertime,auditcontent)

values(:TBID,:OPERATIONSTEP,:STATE,:OPERUSERID,:OPERUSERNAME,sysdate,:AUDITCONTENT);");

sql.Append(@" end; ");

conn.Open();

cmd.CommandText = sql.ToString();

foreach (TblMetaDeTableMasterEntity tableEntity in tableEntitys)

{

cmd.Parameters.Clear();

parametertb[0].Value = (int)ETableState.TestRunReject;

parametertb[1].Value = tableEntity.TBID;

TblMetaDeAuditDetailEntity auditDetail = auditDetails.Where(t => t.TbID == tableEntity.TBID).SingleOrDefault();

if (auditDetail != null)

{

parametertb[2].Value = auditDetail.OperationStep;

parametertb[3].Value = auditDetail.OperUserID;

parametertb[4].Value = auditDetail.OperUserName;

parametertb[5].Value = auditDetail.AuditContent;

}

else

{

parametertb[2].Value = (int)EAuditDetail.TestRunReject;

parametertb[3].Value = "";

parametertb[4].Value = "";

parametertb[5].Value = "";

}

//parametertb[6].Value = DateTime.Now;

parametertb[6].Value = tableEntity.UpdateUserId;

cmd.Parameters.AddRange(parametertb);

cmd.ExecuteNonQuery();

}

result.IsSuccessed = true;

}

catch (Exception ex)

{

result.IsSuccessed = false;

result.Message = ex.Message;

throw ex;

}

finally

{

if (conn != null)

{

conn.Close();

conn.Dispose();

}

}

return result;

}

#endregion

#region 正式部署，更新表状态为运行中

/// <summary>

/// 更新表状态为运行中

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public OperationResult UpdateTableStateToRunning(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

OperationResult result = new OperationResult();

StringBuilder sql = new StringBuilder();

string connectionString = ConnectionStringManager.ETLDesign.ToString();

OracleParameter[] parametertb = new OracleParameter[]

{

new OracleParameter(":STATE",OracleType.Number),

new OracleParameter(":TBID",OracleType.VarChar),

new OracleParameter(":OPERATIONSTEP",OracleType.Number),

new OracleParameter(":OPERUSERID",OracleType.VarChar),

new OracleParameter(":OPERUSERNAME",OracleType.VarChar),

new OracleParameter(":AUDITCONTENT",OracleType.VarChar),

//new OracleParameter(":DATE",OracleType.DateTime),

new OracleParameter(":UpdateUserId",OracleType.VarChar)

};

OracleConnection conn = new OracleConnection(connectionString);

OracleCommand cmd = conn.CreateCommand();

try

{

sql = new StringBuilder();

sql.Append(@"begin update tbl\_meta\_de\_tablemaster set preState=state,state=:State,UpdateUserId =:UpdateUserId,UpdateDate =sysdate,description=:AUDITCONTENT ");

sql.Append(@" where tbid=:tbid;");

sql.Append(@" insert into tbl\_meta\_de\_AuditDetail(tbid, operationstep,state, operuserid,operusername,opertime,auditcontent)

values(:TBID,:OPERATIONSTEP,:STATE,:OPERUSERID,:OPERUSERNAME,sysdate,:AUDITCONTENT);");

sql.Append(@" end; ");

conn.Open();

cmd.CommandText = sql.ToString();

foreach (TblMetaDeTableMasterEntity tableEntity in tableEntitys)

{

cmd.Parameters.Clear();

parametertb[0].Value = (int)ETableState.Running;

parametertb[1].Value = tableEntity.TBID;

TblMetaDeAuditDetailEntity auditDetail = auditDetails.Where(t => t.TbID == tableEntity.TBID).SingleOrDefault();

if (auditDetail != null)

{

parametertb[2].Value = auditDetail.OperationStep;

parametertb[3].Value = auditDetail.OperUserID;

parametertb[4].Value = auditDetail.OperUserName;

parametertb[5].Value = auditDetail.AuditContent;

}

else

{

parametertb[2].Value = (int)EAuditDetail.Running;

parametertb[3].Value = "";

parametertb[4].Value = "";

parametertb[5].Value = "";

}

//parametertb[6].Value = DateTime.Now;

parametertb[6].Value = tableEntity.UpdateUserId;

cmd.Parameters.AddRange(parametertb);

cmd.ExecuteNonQuery();

}

result.IsSuccessed = true;

}

catch (Exception ex)

{

result.IsSuccessed = false;

result.Message = ex.Message;

throw ex;

}

finally

{

if (cmd != null)

{

cmd.Dispose();

}

if (conn != null)

{

conn.Close();

conn.Dispose();

}

}

return result;

}

#endregion

#region 生成元数据到正式环境

public OperationResult ProduceMetaToRun(List<GenerateMetaDataEntity> tables)

{

OperationResult oResult = new OperationResult();

string connectionString = ConnectionStringManager.MetaDataRun;

using (OracleConnection conn = new OracleConnection(connectionString))

{

conn.Open();

OracleTransaction trans = conn.BeginTransaction();

try

{

foreach (GenerateMetaDataEntity table in tables)

{

oResult = GenerateTableMetaDataToRun(table, conn, trans);

oResult = GenerateFieldMetaDataToRun(table, conn, trans);

oResult = GenerateDatabaseMetaDataToRun(table, conn, trans);

oResult = GenerateEnCenterMetaDataToRun(table, conn, trans);

}

trans.Commit();

oResult.IsSuccessed = true;

}

catch (Exception ex)

{

trans.Rollback();

oResult.IsSuccessed = false;

oResult.Message = ex.Message;

}

conn.Close();

conn.Dispose();

}

return oResult;

}

/// <summary>

/// 获得数据库系统值

/// </summary>

/// <param name="\_Seq"></param>

/// <returns></returns>

public int GetSeqNumFromTestRun(string \_Seq)

{

string connectionString = ConnectionStringManager.MetaDataTestRun;

StringBuilder sql = new StringBuilder("select " + \_Seq + " from dual");

return Convert.ToInt32(OracleDbHelper.ExecuteScalar(connectionString, CommandType.Text, sql.ToString(), null));

}

/// <summary>

/// 从正式环境获得数据库系统值

/// </summary>

/// <param name="\_Seq"></param>

/// <returns></returns>

public int GetSeqNumFromRun(string \_Seq)

{

string connectionString = ConnectionStringManager.MetaDataRun;

StringBuilder sql = new StringBuilder("select " + \_Seq + " from dual");

return Convert.ToInt32(OracleDbHelper.ExecuteScalar(connectionString, CommandType.Text, sql.ToString(), null));

}

/// <summary>

/// 生成表元元数据

/// </summary>

/// <param name="table"></param>

/// <returns></returns>

public OperationResult GenerateTableMetaDataToRun(GenerateMetaDataEntity table, OracleConnection conn, OracleTransaction trans)

{

OperationResult oResult = new OperationResult();

try

{

#region 同步表元

string strSql = string.Format(@"MERGE INTO tbl\_meta\_tablemaster t

USING (SELECT Tbtype,Phyiscname, Chiname,Engname, Startdate,

Enddate,Chidesc, Engdesc,Score,Bupdatestate,Dbid,

Bstatedesc,Ordernum,Updatetime,tbid

FROM tbl\_meta\_tablemaster{0}

WHERE tbid = :tbid) S

ON (T.tbid = S.tbid)

WHEN MATCHED THEN

UPDATE

SET T.Tbtype = S.Tbtype,

T.Phyiscname = S.Phyiscname,

T.Chiname = S.Chiname,

T.Engname = S.Engname,

T.Startdate = S.Startdate,

T.Enddate = S.Enddate,

T.Chidesc = S.Chidesc,

T.Engdesc = S.Engdesc,

T.Score = S.Score,

T.Bupdatestate = S.Bupdatestate,

T.Dbid = S.Dbid,

T.Bstatedesc = S.Bstatedesc,

T.Ordernum = S.Ordernum,

T.UPDATETIME = s.updatetime

WHERE t.tbid = :tbid

WHEN NOT MATCHED THEN INSERT(Tbid, Tbtype, Phyiscname, Chiname, Engname, Startdate, Enddate, Chidesc, Engdesc, Score, Bupdatestate, Dbid, Bstatedesc, Ordernum, Updatetime)

VALUES(S.Tbid, S.Tbtype,S.Phyiscname, S.Chiname, S.Engname, S.Startdate, S.Enddate, S.Chidesc, S.Engdesc, S.Score, S.Bupdatestate, S.Dbid, S.Bstatedesc, S.Ordernum, s.updatetime)

",ConnectionStringManager.DesignDbLinkName);

#endregion

OracleParameter[] param = new OracleParameter[]

{

new OracleParameter(":TBID",OracleType.VarChar)

};

param[0].Value = table.Tbid;

int nSeq = GetSeqNumFromRun("SEQ\_META\_TABLECHECK.NEXTVAL");

string strSqlTableCheck = string.Format(@"insert when (not exists(select tbid from TBL\_META\_TABLECHECK where tbid=:tbid)) then

into TBL\_META\_TABLECHECK (TBSID,TBID,CREATEDATE,ACTIONTYPE,CHECKTYPECODE,CREATEUSER)

select {1},:tbid,SYSDATE,'ADD',2,'{0}' from dual", table.UserId, nSeq);

using (OracleCommand cmd = conn.CreateCommand())

{

cmd.Transaction = trans;

cmd.CommandType = CommandType.Text;

cmd.CommandText = strSql;

cmd.Parameters.AddRange(param);

cmd.ExecuteNonQuery();

cmd.CommandText = strSqlTableCheck;

cmd.ExecuteNonQuery();

cmd.Dispose();

//OracleDbHelper.ExecuteNonQuery(connectionString, CommandType.Text, strSql, param);

}

oResult.IsSuccessed = true;

return oResult;

}

catch (Exception ex)

{

throw ex;

}

}

/// <summary>

/// 生成字段元元数据

/// </summary>

/// <param name="table"></param>

/// <returns></returns>

public OperationResult GenerateFieldMetaDataToRun(GenerateMetaDataEntity table, OracleConnection conn, OracleTransaction trans)

{

OperationResult oResult = new OperationResult();

try

{

#region 生成字段

string strSql = string.Format(@"MERGE INTO tbl\_meta\_fieldMaster t

USING (SELECT Phyiscname,

Chiname,

Engname,

Ispk,

Derivefrom,

Isnull,

Interpret,

Chidesc,

Engdesc,

Remark,

Bupdatestate,

Tbid,

Fldtypeid,

Unitid,

Languagetype,

Intodesc,

Bstatedesc,

Codingid,

Actiontype,

case when Isdelete=1 then 1 else NULL END Isdelete,

Ordernum,

Islimit,

Fieldtypename,

Createtime,

Fldid

FROM tbl\_meta\_FieldMaster{0}

WHERE tbid = :tbid) S

ON (T.Fldid = S.Fldid)

WHEN MATCHED THEN

UPDATE

SET T.Phyiscname = S.Phyiscname,

T.Chiname = S.Chiname,

T.Engname = S.Engname,

T.Ispk = S.Ispk,

T.Derivefrom = S.Derivefrom,

T.Isnull = S.Isnull,

T.Interpret = S.Interpret,

T.Chidesc = S.Chidesc,

T.Engdesc = S.Engdesc,

T.Remark = S.Remark,

T.Bupdatestate = S.Bupdatestate,

T.Tbid = S.Tbid,

T.Fldtypeid = S.Fldtypeid,

T.Unitid = S.Unitid,

T.Languagetype = S.Languagetype,

T.Intodesc = S.Intodesc,

T.Bstatedesc = S.Bstatedesc,

T.Codingid = S.Codingid,

T.Actiontype = S.Actiontype,

T.Isdelete = S.Isdelete,

T.Ordernum = S.Ordernum,

T.Islimit = S.Islimit,

T.Fieldtypename = S.Fieldtypename

WHERE t.tbid = :tbid

WHEN NOT MATCHED THEN

INSERT(Fldid, Phyiscname, Chiname, Engname, Ispk, Derivefrom, Isnull, Interpret, Chidesc, Engdesc, Remark, Bupdatestate, Tbid, Fldtypeid, Unitid,

Languagetype, Intodesc, Bstatedesc, Codingid, Actiontype, Isdelete, Ordernum, Islimit, Fieldtypename, Createtime)

VALUES(S.Fldid, S.Phyiscname,S.Chiname, S.Engname, S.Ispk, S.Derivefrom, S.Isnull, S.Interpret, S.Chidesc, S.Engdesc, S.Remark, S.Bupdatestate, S.Tbid, S.Fldtypeid, S.Unitid,

S.Languagetype, S.Intodesc, S.Bstatedesc, S.Codingid, S.Actiontype, S.Isdelete, S.Ordernum, S.Islimit, S.Fieldtypename, SYSDATE) ", ConnectionStringManager.DesignDbLinkName);

#endregion

OracleParameter[] param = new OracleParameter[]

{

new OracleParameter(":TBID",OracleType.VarChar)

};

param[0].Value = table.Tbid;

using (OracleCommand cmd = conn.CreateCommand())

{

cmd.Transaction = trans;

cmd.CommandType = CommandType.Text;

cmd.CommandText = strSql;

cmd.Parameters.AddRange(param);

cmd.ExecuteNonQuery();

cmd.Dispose();

//OracleDbHelper.ExecuteNonQuery(connectionString, CommandType.Text, strSql, param);

}

oResult.IsSuccessed = true;

return oResult;

}

catch (Exception ex)

{

throw ex;

}

}

/// <summary>

/// 生成主题域元数据

/// </summary>

/// <param name="table"></param>

/// <returns></returns>

public OperationResult GenerateDatabaseMetaDataToRun(GenerateMetaDataEntity table, OracleConnection conn, OracleTransaction trans)

{

OperationResult oResult = new OperationResult();

try

{

#region 生成主题域信息

string strSql = string.Format(@"MERGE INTO tbl\_meta\_databaseMaster t

USING (SELECT Name,

Typeid,

Chidesc,

Engdesc,

Startdate,

Enddate,

Updateserid,

Updatetime,

DBID,type,parentId

from tbl\_meta\_databaseMaster{0} db

CONNECT BY PRIOR parentid = dbid

START WITH dbid in (select dbid

from tbl\_meta\_tableMaster{0}

where tbid = :tbid)) S

ON (T.DBID = S.DBID)

WHEN MATCHED THEN

UPDATE

SET T.Name = S.Name,

T.Typeid = S.Typeid,

T.Chidesc = S.Chidesc,

T.Engdesc = S.Engdesc,

T.Startdate = S.Startdate,

T.Enddate = S.Enddate,

T.Updateserid = S.Updateserid,

T.Updatetime = S.Updatetime,

T.Type =S.Type,

T.parentId=S.parentId

WHERE dbid in (SELECT DBID

from tbl\_meta\_databaseMaster db

CONNECT BY PRIOR

parentid = dbid

START WITH dbid IN

(select dbid

from tbl\_meta\_tableMaster

where tbid =

:tbid))

WHEN NOT MATCHED THEN INSERT(Dbid, Name, Typeid, Chidesc, Engdesc, Startdate, Enddate, Updateserid, Updatetime,Type,parentId)

VALUES(S.Dbid, S.Name, S.Typeid, S.Chidesc, S.Engdesc, S.Startdate, S.Enddate, S.Updateserid, s.Updatetime,S.Type,S.ParentId) ",ConnectionStringManager.DesignDbLinkName);

#endregion

OracleParameter[] param = new OracleParameter[]

{

new OracleParameter(":TBID",OracleType.VarChar)

};

param[0].Value = table.Tbid;

using (OracleCommand cmd = conn.CreateCommand())

{

cmd.Transaction = trans;

cmd.CommandType = CommandType.Text;

cmd.CommandText = strSql;

cmd.Parameters.AddRange(param);

cmd.ExecuteNonQuery();

cmd.Dispose();

//OracleDbHelper.ExecuteNonQuery(connectionString, CommandType.Text, strSql, param);

}

oResult.IsSuccessed = true;

return oResult;

}

catch (Exception ex)

{

throw ex;

}

}

/// <summary>

/// 生成英文库元数据

/// </summary>

/// <param name="\_tbid"></param>

/// <returns></returns>

public OperationResult GenerateEnCenterMetaDataToRun(GenerateMetaDataEntity table, OracleConnection conn, OracleTransaction trans)

{

OperationResult oResult = new OperationResult();

string connectionString = ConnectionStringManager.MetaDataRun;//设计库元数据

try

{

#region 生成英文库元数据

if (table.IsNeedGenerate)

{

int result = 0;

string procName = "SP\_TBL\_META\_TABLE\_TRANS";

using (DbCommand \_command = conn.CreateCommand())

{

\_command.Transaction = trans;

\_command.CommandText = procName;

\_command.CommandType = CommandType.StoredProcedure;

\_command.Parameters.Add(new OracleParameter("TYPEID", table.BTypeId));

\_command.Parameters.Add(new OracleParameter("TBID", table.Tbid));

\_command.Parameters.Add(new OracleParameter("LANGTYPE", table.LangType));

\_command.Parameters.Add(new OracleParameter("LANGCODE", table.LangCode));

\_command.Parameters.Add(new OracleParameter("USERID", table.UserId));

\_command.Parameters.Add(new OracleParameter("RESCOUNT", 10));

\_command.ExecuteNonQuery();

result = Convert.ToInt32(\_command.Parameters["RESCOUNT"].Value.ToString());

\_command.Dispose();

}

if (result <= 0)

throw new Exception("表" + table.TableName + "生成英文库元数据时出错！");

}

#endregion

oResult.IsSuccessed = true;

}

catch (Exception ex)

{

oResult.IsSuccessed = false;

oResult.Message = ex.Message;

return oResult;

}

return oResult;

}

#endregion

#region 生成元数据到试运行

public OperationResult ProduceMetaToTestRun(List<GenerateMetaDataEntity> tables)

{

OperationResult oResult = new OperationResult();

string connectionString = ConnectionStringManager.MetaDataTestRun;

using (OracleConnection conn = new OracleConnection(connectionString))

{

conn.Open();

OracleTransaction trans = conn.BeginTransaction();

try

{

foreach (GenerateMetaDataEntity table in tables)

{

oResult = GenerateTableMetaDataToTestRun(table, conn, trans);

oResult = GenerateFieldMetaDataToTestRun(table, conn, trans);

oResult = GenerateDatabaseMetaDataToTestRun(table, conn, trans);

oResult = GenerateEnCenterMetaDataToTestRun(table, conn, trans);

}

trans.Commit();

oResult.IsSuccessed = true;

}

catch (Exception ex)

{

trans.Rollback();

oResult.IsSuccessed = false;

oResult.Message = ex.Message;

}

conn.Close();

conn.Dispose();

}

return oResult;

}

/// <summary>

/// 生成表元元数据

/// </summary>

/// <param name="table"></param>

/// <returns></returns>

public OperationResult GenerateTableMetaDataToTestRun(GenerateMetaDataEntity table, OracleConnection conn, OracleTransaction trans)

{

OperationResult oResult = new OperationResult();

try

{

#region 同步表元

string strSql = string.Format(@"MERGE INTO tbl\_meta\_tablemaster t

USING (SELECT Tbtype,Phyiscname, Chiname,Engname, Startdate,

Enddate,Chidesc, Engdesc,Score,Bupdatestate,Dbid,

Bstatedesc,Ordernum,Updatetime,tbid

FROM tbl\_meta\_tablemaster{0}

WHERE tbid = :tbid) S

ON (T.tbid = S.tbid)

WHEN MATCHED THEN

UPDATE

SET T.Tbtype = S.Tbtype,

T.Phyiscname = S.Phyiscname,

T.Chiname = S.Chiname,

T.Engname = S.Engname,

T.Startdate = S.Startdate,

T.Enddate = S.Enddate,

T.Chidesc = S.Chidesc,

T.Engdesc = S.Engdesc,

T.Score = S.Score,

T.Bupdatestate = S.Bupdatestate,

T.Dbid = S.Dbid,

T.Bstatedesc = S.Bstatedesc,

T.Ordernum = S.Ordernum,

T.UPDATETIME = s.updatetime

WHERE t.tbid = :tbid

WHEN NOT MATCHED THEN INSERT(Tbid, Tbtype, Phyiscname, Chiname, Engname, Startdate, Enddate, Chidesc, Engdesc, Score, Bupdatestate, Dbid, Bstatedesc, Ordernum, Updatetime)

VALUES(S.Tbid, S.Tbtype,S.Phyiscname, S.Chiname, S.Engname, S.Startdate, S.Enddate, S.Chidesc, S.Engdesc, S.Score, S.Bupdatestate, S.Dbid, S.Bstatedesc, S.Ordernum, s.updatetime)

",ConnectionStringManager.DesignDbLinkName);

#endregion

OracleParameter[] param = new OracleParameter[]

{

new OracleParameter(":TBID",OracleType.VarChar)

};

param[0].Value = table.Tbid;

int nSeq = GetSeqNumFromTestRun("SEQ\_META\_TABLECHECK.NEXTVAL");

string strSqlTableCheck = string.Format(@"insert when (not exists(select tbid from TBL\_META\_TABLECHECK where tbid=:tbid)) then

into TBL\_META\_TABLECHECK (TBSID,TBID,CREATEDATE,ACTIONTYPE,CHECKTYPECODE,CREATEUSER)

select {1},:tbid,SYSDATE,'ADD',2,'{0}' from dual", table.UserId, nSeq);

using (OracleCommand cmd = conn.CreateCommand())

{

cmd.Transaction = trans;

cmd.CommandType = CommandType.Text;

cmd.CommandText = strSql;

cmd.Parameters.AddRange(param);

cmd.ExecuteNonQuery();

cmd.CommandText = strSqlTableCheck;

cmd.ExecuteNonQuery();

cmd.Dispose();

//OracleDbHelper.ExecuteNonQuery(connectionString, CommandType.Text, strSql, param);

}

oResult.IsSuccessed = true;

return oResult;

}

catch (Exception ex)

{

throw ex;

}

}

/// <summary>

/// 生成字段元元数据

/// </summary>

/// <param name="table"></param>

/// <returns></returns>

public OperationResult GenerateFieldMetaDataToTestRun(GenerateMetaDataEntity table, OracleConnection conn, OracleTransaction trans)

{

OperationResult oResult = new OperationResult();

try

{

#region 生成字段

string strSql = string.Format(@"MERGE INTO tbl\_meta\_fieldMaster t

USING (SELECT Phyiscname,

Chiname,

Engname,

Ispk,

Derivefrom,

Isnull,

Interpret,

Chidesc,

Engdesc,

Remark,

Bupdatestate,

Tbid,

Fldtypeid,

Unitid,

Languagetype,

Intodesc,

Bstatedesc,

Codingid,

Actiontype,

Isdelete,

Ordernum,

Islimit,

Fieldtypename,

Createtime,

Fldid

FROM tbl\_meta\_FieldMaster{0}

WHERE tbid = :tbid) S

ON (T.Fldid = S.Fldid)

WHEN MATCHED THEN

UPDATE

SET T.Phyiscname = S.Phyiscname,

T.Chiname = S.Chiname,

T.Engname = S.Engname,

T.Ispk = S.Ispk,

T.Derivefrom = S.Derivefrom,

T.Isnull = S.Isnull,

T.Interpret = S.Interpret,

T.Chidesc = S.Chidesc,

T.Engdesc = S.Engdesc,

T.Remark = S.Remark,

T.Bupdatestate = S.Bupdatestate,

T.Tbid = S.Tbid,

T.Fldtypeid = S.Fldtypeid,

T.Unitid = S.Unitid,

T.Languagetype = S.Languagetype,

T.Intodesc = S.Intodesc,

T.Bstatedesc = S.Bstatedesc,

T.Codingid = S.Codingid,

T.Actiontype = S.Actiontype,

T.Isdelete = S.Isdelete,

T.Ordernum = S.Ordernum,

T.Islimit = S.Islimit,

T.Fieldtypename = S.Fieldtypename

WHERE t.tbid = :tbid

WHEN NOT MATCHED THEN

INSERT(Fldid, Phyiscname, Chiname, Engname, Ispk, Derivefrom, Isnull, Interpret, Chidesc, Engdesc, Remark, Bupdatestate, Tbid, Fldtypeid, Unitid,

Languagetype, Intodesc, Bstatedesc, Codingid, Actiontype, Isdelete, Ordernum, Islimit, Fieldtypename, Createtime)

VALUES(S.Fldid, S.Phyiscname,S.Chiname, S.Engname, S.Ispk, S.Derivefrom, S.Isnull, S.Interpret, S.Chidesc, S.Engdesc, S.Remark, S.Bupdatestate, S.Tbid, S.Fldtypeid, S.Unitid,

S.Languagetype, S.Intodesc, S.Bstatedesc, S.Codingid, S.Actiontype, S.Isdelete, S.Ordernum, S.Islimit, S.Fieldtypename, SYSDATE) ",ConnectionStringManager.DesignDbLinkName);

#endregion

OracleParameter[] param = new OracleParameter[]

{

new OracleParameter(":TBID",OracleType.VarChar)

};

param[0].Value = table.Tbid;

using (OracleCommand cmd = conn.CreateCommand())

{

cmd.Transaction = trans;

cmd.CommandType = CommandType.Text;

cmd.CommandText = strSql;

cmd.Parameters.AddRange(param);

cmd.ExecuteNonQuery();

cmd.Dispose();

//OracleDbHelper.ExecuteNonQuery(connectionString, CommandType.Text, strSql, param);

}

oResult.IsSuccessed = true;

return oResult;

}

catch (Exception ex)

{

throw ex;

}

}

/// <summary>

/// 生成主题域元数据

/// </summary>

/// <param name="table"></param>

/// <returns></returns>

public OperationResult GenerateDatabaseMetaDataToTestRun(GenerateMetaDataEntity table, OracleConnection conn, OracleTransaction trans)

{

OperationResult oResult = new OperationResult();

try

{

#region 生成主题域信息

string strSql =string.Format( @"MERGE INTO tbl\_meta\_databaseMaster t

USING (SELECT Name,

Typeid,

Chidesc,

Engdesc,

Startdate,

Enddate,

Updateserid,

Updatetime,

DBID,type,parentId

from tbl\_meta\_databaseMaster{0} db

CONNECT BY PRIOR parentid = dbid

START WITH dbid in (select dbid

from tbl\_meta\_tableMaster{0}

where tbid = :tbid)) S

ON (T.DBID = S.DBID)

WHEN MATCHED THEN

UPDATE

SET T.Name = S.Name,

T.Typeid = S.Typeid,

T.Chidesc = S.Chidesc,

T.Engdesc = S.Engdesc,

T.Startdate = S.Startdate,

T.Enddate = S.Enddate,

T.Updateserid = S.Updateserid,

T.Updatetime = S.Updatetime,

T.Type =S.Type,

T.parentId=S.parentId

WHERE dbid in (SELECT DBID

from tbl\_meta\_databaseMaster db

CONNECT BY PRIOR

parentid = dbid

START WITH dbid IN

(select dbid

from tbl\_meta\_tableMaster

where tbid =

:tbid))

WHEN NOT MATCHED THEN INSERT(Dbid, Name, Typeid, Chidesc, Engdesc, Startdate, Enddate, Updateserid, Updatetime,Type,parentId)

VALUES(S.Dbid, S.Name, S.Typeid, S.Chidesc, S.Engdesc, S.Startdate, S.Enddate, S.Updateserid, s.Updatetime,S.Type,S.ParentId) ",ConnectionStringManager.DesignDbLinkName);

#endregion

OracleParameter[] param = new OracleParameter[]

{

new OracleParameter(":TBID",OracleType.VarChar)

};

param[0].Value = table.Tbid;

using (OracleCommand cmd = conn.CreateCommand())

{

cmd.Transaction = trans;

cmd.CommandType = CommandType.Text;

cmd.CommandText = strSql;

cmd.Parameters.AddRange(param);

cmd.ExecuteNonQuery();

cmd.Dispose();

//OracleDbHelper.ExecuteNonQuery(connectionString, CommandType.Text, strSql, param);

}

oResult.IsSuccessed = true;

return oResult;

}

catch (Exception ex)

{

throw ex;

}

}

/// <summary>

/// 生成英文库元数据

/// </summary>

/// <param name="\_tbid"></param>

/// <returns></returns>

public OperationResult GenerateEnCenterMetaDataToTestRun(GenerateMetaDataEntity table, OracleConnection conn, OracleTransaction trans)

{

OperationResult oResult = new OperationResult();

try

{

#region 生成英文库元数据

if (table.IsNeedGenerate)

{

int result = 0;

string procName = "SP\_TBL\_META\_TABLE\_TRANS";

using (DbCommand \_command = conn.CreateCommand())

{

\_command.Transaction = trans;

\_command.CommandText = procName;

\_command.CommandType = CommandType.StoredProcedure;

\_command.Parameters.Add(new OracleParameter("TYPEID", table.BTypeId));

\_command.Parameters.Add(new OracleParameter("TBID", table.Tbid));

\_command.Parameters.Add(new OracleParameter("LANGTYPE", table.LangType));

\_command.Parameters.Add(new OracleParameter("LANGCODE", table.LangCode));

\_command.Parameters.Add(new OracleParameter("USERID", table.UserId));

\_command.Parameters.Add(new OracleParameter("RESCOUNT", 10));

\_command.ExecuteNonQuery();

result = Convert.ToInt32(\_command.Parameters["RESCOUNT"].Value.ToString());

\_command.Dispose();

}

if (result <= 0)

throw new Exception("表" + table.TableName + "生成英文库元数据时出错！");

}

#endregion

oResult.IsSuccessed = true;

}

catch (Exception ex)

{

oResult.IsSuccessed = false;

oResult.Message = ex.Message;

return oResult;

}

return oResult;

}

#endregion

#endregion

#region ETL开发人员操作界面

/// <summary>

/// 开发人员查询表信息

/// </summary>

/// <param name="tableEntitys"></param>

/// <returns></returns>

public PagingResult QueryDevelopmentTables(AuditQueryParam para)

{

PagingResult pagingResult = new PagingResult();

StringBuilder sql = new StringBuilder();

StringBuilder sqlconnect = new StringBuilder();

StringBuilder sqlWhere = new StringBuilder();

List<OracleParameter> querySQLParameterList = new List<OracleParameter>();

sql.Append(@" select tb.docid,doc.docname,buss.business,db.dbid,db.name,tb.tbid,

tb.phyiscname,tb.chiname,

case tb.operationtype when 0 then '新增' when 1 then '修改' when 2 then '删除' when 3 then '提升' end as operationtype,

tb.state,user1.username as createuser,

to\_char(tb.createdate,'yyyy-mm-dd hh24:mi:ss') as createdate,

user2.username as updateuser,

to\_char(tb.updatedate,'yyyy-mm-dd hh24:mi:ss') as updatedate,

tb.developmentuserid,uDev.userName as DevelopmentUserName,tb.testuserid,

user3.username as testuser,tb.description ");

if (!string.IsNullOrEmpty(para.KeyWord))

{

sql.Append(@" , case when upper(db.dbid) like :querytext escape '/' then length(db.dbid)

when upper(db.name) like :querytext escape '/' then length(db.name)

when upper(tb.tbid) like :querytext escape '/' then length(tb.tbid)

when upper(tb.phyiscname) like :querytext escape '/' then length(tb.phyiscname)

when upper(tb.chiname) like :querytext escape '/' then length(tb.chiname) end as FieldLen

");

}

sql.Append(@" from tbl\_meta\_de\_tablemaster tb ");

sqlWhere.Append("where 1=1 ");

if (Convert.ToInt32(para.TableState) >= 0)

{

sqlWhere.Append(" and tb.state = :tbstate");

querySQLParameterList.Add(new OracleParameter(":tbstate", para.TableState));

}

if (!string.IsNullOrEmpty(para.DevelopmentUserId))

{

sqlWhere.Append(" and tb.developmentuserid = :developmentuserid");

querySQLParameterList.Add(new OracleParameter(":developmentuserid", para.DevelopmentUserId));

}

if (!string.IsNullOrEmpty(para.TestUserId))

{

sqlWhere.Append(" and tb.testuserid = :testUserId");

querySQLParameterList.Add(new OracleParameter(":testUserId", para.TestUserId));

}

if (!string.IsNullOrEmpty(para.KeyWord))

{

sqlWhere.Append(@" and (upper(db.dbid) like :querytext escape '/' or upper(db.name) like :querytext escape '/'

or upper(tb.tbid) like :querytext escape '/' or upper(tb.phyiscname) like :querytext escape '/'

or upper(tb.chiname) like :querytext escape '/')

");

querySQLParameterList.Add(new OracleParameter(":querytext", "%" + para.KeyWord.ToUpper() + "%"));

}

sqlconnect.Append("left join tbl\_meta\_de\_databasemaster db on tb.dbid=db.dbid ");

sqlconnect.Append("left join tbl\_meta\_de\_document doc on tb.docid=doc.docid ");

sqlconnect.Append("left join tbl\_meta\_de\_businesstype buss on db.typeid=buss.btypeid ");

sqlconnect.Append("left join tbl\_um\_user user1 on tb.createuserid=user1.userid ");

sqlconnect.Append("left join tbl\_um\_user user2 on tb.updateuserid=user2.userid ");

sqlconnect.Append("left join tbl\_um\_user user3 on tb.testuserid=user3.userid ");

sqlconnect.Append("left join tbl\_um\_user uDev on tb.developmentuserid=uDev.userid ");

string strOrderBy = "";

if (!string.IsNullOrEmpty(para.KeyWord))

{

strOrderBy = " order by FieldLen asc,updatedate desc";

}

else

strOrderBy = " order by updatedate desc";

string executeSql = sql.ToString() + sqlconnect.ToString() + sqlWhere.ToString() + strOrderBy;

string pageSql = string.Format(@"select \* from(select t.\*,rownum rn from({0}) t where rownum<={1}) where rn>{2} ", executeSql, para.PageSize \* para.PageIndex, para.PageSize \* (para.PageIndex - 1));

pagingResult.DataSet = OracleDbHelper.LoadDataSetQuery(ConnectionStringManager.ETLDesign, CommandType.Text, pageSql, querySQLParameterList.ToArray());

string countSql = string.Format(@"select count(1) from tbl\_meta\_de\_tablemaster tb {0}", sqlconnect.ToString() + sqlWhere.ToString());

pagingResult.RowCount = Convert.ToInt32(OracleDbHelper.ExecuteScalar(ConnectionStringManager.ETLDesign, CommandType.Text, countSql, querySQLParameterList.ToArray()));

return pagingResult;

}

/// <summary>

/// 操作记录查询

/// </summary>

/// <param name="tbid"></param>

/// <returns></returns>

public DataSet QueryOperationList(string tbid)

{

string strSql = @" select ad.tbid ,t.phyiscname as TABLEPHYSICNAME,t.chiname as TABLECHNAME ,ad.operationStep as OPERATIONTYPENAME ,

u.username as OperationUserName, ad.opertime as OPERATIONTIME,ad.AUDITCONTENT

from tbl\_meta\_de\_auditdetail ad left join tbl\_meta\_de\_tablemaster t on ad.tbid=t.tbid

left join tbl\_um\_user u on ad.operuserid =u.userid

where ad.tbid =:tbid order by opertime desc";

List<OracleParameter> listSQLParam = new List<OracleParameter>();

listSQLParam.Add(new OracleParameter(":tbid", tbid));

DataSet ds = OracleDbHelper.LoadDataSetQuery(ConnectionStringManager.ETLDesign, CommandType.Text, strSql, listSQLParam.ToArray());

return ds;

}

/// <summary>

/// 开发人员不通过

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public OperationResult DevelopmentReject(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

OperationResult result = new OperationResult();

StringBuilder sql = new StringBuilder();

string connectionString = ConnectionStringManager.ETLDesign.ToString();

OracleParameter[] parametertb = new OracleParameter[]

{

new OracleParameter(":State",OracleType.Number),

new OracleParameter(":tbid",OracleType.VarChar),

new OracleParameter(":OPERATIONSTEP",OracleType.Number),

new OracleParameter(":OPERUSERID",OracleType.VarChar),

new OracleParameter(":OPERUSERNAME",OracleType.VarChar),

new OracleParameter(":AUDITCONTENT",OracleType.VarChar)

//new OracleParameter(":Date",OracleType.DateTime)

};

OracleConnection conn = new OracleConnection(connectionString);

OracleCommand cmd = conn.CreateCommand();

try

{

sql = new StringBuilder();

sql.Append(@"begin update tbl\_meta\_de\_tablemaster set preState=state,state=:State,description='【开发】：'||:AUDITCONTENT,updateuserId=:OPERUSERID,updatedate=sysdate ");

sql.Append(@" where tbid=:tbid;");

sql.Append(@" insert into tbl\_meta\_de\_AuditDetail(tbid, operationstep,state, operuserid,operUserName,OPERTIME,auditcontent)

values(:TBID,:OPERATIONSTEP,:state,:OPERUSERID,:OPERUSERNAME,sysdate,:AUDITCONTENT);");

sql.Append(@" end; ");

conn.Open();

cmd.CommandText = sql.ToString();

foreach (TblMetaDeTableMasterEntity tableEntity in tableEntitys)

{

cmd.Parameters.Clear();

parametertb[0].Value = (int)ETableState.DevelopReject;

parametertb[1].Value = tableEntity.TBID;

TblMetaDeAuditDetailEntity auditDetail = auditDetails.Where(t => t.TbID == tableEntity.TBID).SingleOrDefault();

if (auditDetail != null)

{

parametertb[2].Value = auditDetail.OperationStep;

parametertb[3].Value = auditDetail.OperUserID;

parametertb[4].Value = auditDetail.OperUserName;

parametertb[5].Value = auditDetail.AuditContent;

}

else

{

parametertb[2].Value = (int)EAuditDetail.DevelopmentReject;

parametertb[3].Value = "";

parametertb[4].Value = "";

parametertb[5].Value ="";

}

//parametertb[6].Value = DateTime.Now;

cmd.Parameters.AddRange(parametertb);

cmd.ExecuteNonQuery();

}

result.IsSuccessed = true;

}

catch (Exception ex)

{

result.IsSuccessed = false;

result.Message = ex.Message;

throw ex;

}

finally

{

if (conn != null)

{

conn.Close();

conn.Dispose();

}

}

return result;

}

/// <summary>

/// 开发提交测试

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public int SubmitTest(List<TblMetaDeTableMasterEntity> lstTable, List<TblMetaDeAuditDetailEntity> lstAuditDetail)

{

int result = 0;

string connectionString = ConnectionStringManager.ETLDesign;

//打开数据库连接

OracleDatabase db = new OracleDatabase(connectionString);

DbConnection conn = db.CreateConnection();

conn.Open();

//开启事务

DbTransaction trans = conn.BeginTransaction();

try

{

//更改表状态

if (lstTable != null && lstTable.Count > 0)

{

ChangeTableStateToTesting(lstTable, conn, trans);

}

//插入表审核记录

if (lstAuditDetail != null && lstAuditDetail.Count > 0)

{

InsertAuditDetail(lstAuditDetail, conn, trans);

}

trans.Commit();

result = 1;

}

catch (Exception ex)

{

//保存失败，执行回滚

trans.Rollback();

result = 0;

}

finally

{

conn.Close();

conn.Dispose();

}

return result;

}

/// <summary>

/// 更改表状态为测试中状态

/// </summary>

/// <param name="lstTable"></param>

/// <param name="conn"></param>

/// <param name="trans"></param>

private bool ChangeTableStateToTesting(List<TblMetaDeTableMasterEntity> lstTable, DbConnection conn, DbTransaction trans)

{

bool result = true;

string sql = string.Empty;

try

{

sql = @"begin update tbl\_meta\_de\_tablemaster

set State = :State,

prestate = :prestate,

TestUserId = :TestUserId,

TestUserName = :TestUserName,

updateuserId =:OPERUSERID,

updatedate =sysdate

where tbid = :tbid; end;";

DbCommand cmd = conn.CreateCommand();

cmd.Transaction = trans;

cmd.CommandText = sql;

OracleParameter[] parameter = new OracleParameter[]{

new OracleParameter(":State",OracleType.Number),

new OracleParameter(":prestate",OracleType.Number),

new OracleParameter(":TestUserId",OracleType.VarChar),

new OracleParameter(":TestUserName",OracleType.VarChar),

new OracleParameter(":tbid",OracleType.VarChar),

new OracleParameter(":OPERUSERID",OracleType.VarChar)

//new OracleParameter(":Date",OracleType.DateTime)

};

foreach (TblMetaDeTableMasterEntity entity in lstTable)

{

cmd.Parameters.Clear();

parameter[0].Value = entity.State;

parameter[1].Value = entity.PreState;

parameter[2].Value = entity.TestUserId;

parameter[3].Value = entity.TestUserName;

parameter[4].Value = entity.TBID;

parameter[5].Value = entity.UpdateUserId;

//parameter[6].Value = DateTime.Now;

cmd.Parameters.AddRange(parameter);

cmd.ExecuteNonQuery();

}

result = true;

}

catch (Exception ex)

{

result = false;

throw ex;

}

return result;

}

#endregion

#region 测试任务管理界面

/// <summary>

/// 测试通过

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public OperationResult TestOK(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

OperationResult result = new OperationResult();

StringBuilder sql = new StringBuilder();

string connectionString = ConnectionStringManager.ETLDesign.ToString();

OracleParameter[] parametertb = new OracleParameter[]

{

new OracleParameter(":State",OracleType.Number),

new OracleParameter(":tbid",OracleType.VarChar),

new OracleParameter(":OPERATIONSTEP",OracleType.Number),

new OracleParameter(":OPERUSERID",OracleType.VarChar),

new OracleParameter(":OPERUSERNAME",OracleType.VarChar)

//new OracleParameter(":Date",OracleType.DateTime)

};

OracleConnection conn = new OracleConnection(connectionString);

conn.Open();

OracleTransaction trans = conn.BeginTransaction();

try

{

sql = new StringBuilder();

sql.Append(@"begin update tbl\_meta\_de\_tablemaster set preState=state,state=:State,updateuserId=:OPERUSERID,updatedate=sysdate ");

sql.Append(@" where tbid=:tbid;");

sql.Append(@" insert into tbl\_meta\_de\_AuditDetail(tbid, operationstep,state, operuserid,operUserName,OPERTIME)

values(:TBID,:OPERATIONSTEP,:state,:OPERUSERID,:OPERUSERNAME,sysdate);");

sql.Append(@" end;");

using(OracleCommand cmd = conn.CreateCommand())

{

cmd.Transaction = trans;

cmd.CommandText = sql.ToString();

foreach (TblMetaDeTableMasterEntity tableEntity in tableEntitys)

{

cmd.Parameters.Clear();

parametertb[0].Value = (int)ETableState.TestPass;

parametertb[1].Value = tableEntity.TBID;

TblMetaDeAuditDetailEntity auditDetail = auditDetails.Where(t => t.TbID == tableEntity.TBID).SingleOrDefault();

if (auditDetail != null)

{

parametertb[2].Value = auditDetail.OperationStep;

parametertb[3].Value = auditDetail.OperUserID;

parametertb[4].Value = auditDetail.OperUserName;

}

else

{

parametertb[2].Value = (int)EAuditDetail.DevelopmentReject;

parametertb[3].Value = "";

parametertb[4].Value = "";

}

//parametertb[5].Value = DateTime.Now;

cmd.Parameters.AddRange(parametertb);

cmd.ExecuteNonQuery();

}

}

trans.Commit();

result.IsSuccessed = true;

}

catch (Exception ex)

{

trans.Rollback();

result.IsSuccessed = false;

result.Message = ex.Message;

throw ex;

}

finally

{

if (conn != null)

{

conn.Close();

conn.Dispose();

}

}

return result;

}

/// <summary>

/// 测试不通过

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public OperationResult TestReject(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

OperationResult result = new OperationResult();

StringBuilder sql = new StringBuilder();

string connectionString = ConnectionStringManager.ETLDesign.ToString();

OracleParameter[] parametertb = new OracleParameter[]

{

new OracleParameter(":State",OracleType.Number),

new OracleParameter(":tbid",OracleType.VarChar),

new OracleParameter(":OPERATIONSTEP",OracleType.Number),

new OracleParameter(":OPERUSERID",OracleType.VarChar),

new OracleParameter(":OPERUSERNAME",OracleType.VarChar),

new OracleParameter(":AUDITCONTENT",OracleType.VarChar)

//new OracleParameter(":Date",OracleType.DateTime)

};

OracleConnection conn = new OracleConnection(connectionString);

conn.Open();

OracleTransaction trans = conn.BeginTransaction();

try

{

sql = new StringBuilder();

sql.Append(@"begin update tbl\_meta\_de\_tablemaster set preState=state,state=:State ,description='【测试】：'||:AUDITCONTENT ,updateuserId=:OPERUSERID,updatedate=sysdate ");

sql.Append(@" where tbid=:tbid;");

sql.Append(@" insert into tbl\_meta\_de\_AuditDetail(tbid, operationstep,state, operuserid,operUserName,OPERTIME,AUDITCONTENT)

values(:TBID,:OPERATIONSTEP,:state,:OPERUSERID,:OPERUSERNAME,sysdate,:AUDITCONTENT);");

sql.Append(@" end;");

using (OracleCommand cmd = conn.CreateCommand())

{

cmd.Transaction = trans;

cmd.CommandText = sql.ToString();

foreach (TblMetaDeTableMasterEntity tableEntity in tableEntitys)

{

cmd.Parameters.Clear();

parametertb[0].Value = (int)ETableState.TestReject;

parametertb[1].Value = tableEntity.TBID;

TblMetaDeAuditDetailEntity auditDetail = auditDetails.Where(t => t.TbID == tableEntity.TBID).SingleOrDefault();

if (auditDetail != null)

{

parametertb[2].Value = auditDetail.OperationStep;

parametertb[3].Value = auditDetail.OperUserID;

parametertb[4].Value = auditDetail.OperUserName;

parametertb[5].Value = auditDetail.AuditContent;

}

else

{

parametertb[2].Value = (int)EAuditDetail.TestReject;

parametertb[3].Value = "";

parametertb[4].Value = "";

parametertb[5].Value = "";

}

//parametertb[6].Value =DateTime.Now ;

cmd.Parameters.AddRange(parametertb);

cmd.ExecuteNonQuery();

}

}

trans.Commit();

result.IsSuccessed = true;

}

catch (Exception ex)

{

if (trans != null)

{

trans.Rollback();

}

result.IsSuccessed = false;

result.Message = ex.Message;

throw ex;

}

finally

{

if (conn != null)

{

conn.Close();

conn.Dispose();

}

}

return result;

}

#endregion

#region 生成元数据后更新表的是否生成元数据标识

/// <summary>

/// 生成元数据后更新表的是否生成元数据标识

/// </summary>

/// <param name="\_tbids"></param>

/// <param name="isGenerateMetaData">isGenerateMetaData 1:生成到设计库，2，生成到试运行环境，3生成到正式环境</param>

/// <returns></returns>

public bool UpateTableIsGenerateData(List<GenerateMetaDataEntity> tableEntitys, int isCreatedMetaData)

{

string connectionString = ConnectionStringManager.ETLDesign;//设计库元数据

OracleConnection conn = new OracleConnection(connectionString);

conn.Open();

try

{

using (OracleCommand cmd = conn.CreateCommand())

{

string strSql = @"update tbl\_meta\_de\_tablemaster set iscreatedmetadata=:IsCreatedMetaData where tbid=:TBID ";

OracleParameter[] param = new OracleParameter[]

{

new OracleParameter(":TBID",OracleType.VarChar),

new OracleParameter(":IsCreatedMetaData",OracleType.VarChar)

};

cmd.CommandText = strSql;

cmd.CommandType = CommandType.Text;

foreach (GenerateMetaDataEntity \_tbid in tableEntitys)

{

cmd.Parameters.Clear();

param[0].Value = \_tbid.Tbid;

param[1].Value = isCreatedMetaData;

cmd.Parameters.AddRange(param);

cmd.ExecuteNonQuery();

}

}

}

catch (Exception ex)

{

return false;

}

finally

{

conn.Close();

conn.Dispose();

}

return true;

}

#endregion

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

using GTA.DC.MetaFactory.IDAL;

using GTA.DataLogCommon;

using GTA.DC.MetaFactory.Entity;

using System.Transactions;

namespace GTA.DC.MetaFactory.BLL

{

public class AuditInformationBLL

{

#region 查阅审核操作的详细信息

/// <summary>

/// 查阅审核操作的详细信息

/// </summary>

/// <param name="\_tbid"></param>

/// <returns></returns>

public DataSet QueryTableCheckMessage(string \_tbid)

{

try

{

IAuditInformationDAL \_tablepromotedal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_tablepromotedal.QueryTableCheckMessage(\_tbid);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"DataSet QueryTableCheckMessage(decimal \_tbid)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw;

}

}

#endregion

#region 项目经理查阅表审核信息

public PagingResult ProjMgrQueryTable(string querytext, int tbstate, int pageIndex, int pageSize)

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.ProjMgrQueryTable(querytext, tbstate, pageIndex, pageSize);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"PagingResult ProjMgrQueryTable(string querytext, int tbstate, int pageIndex, int pageSize)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw;

}

}

#endregion

#region 抽取生成元数据

/// <summary>

/// 抽取生成元数据

/// </summary>

/// <param name="\_tbid"></param>

/// <returns></returns>

public bool ProduceMeta(Dictionary<string, string> \_tbids)

{

try

{

IAuditInformationDAL \_tablepromotedal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_tablepromotedal.ProduceMeta(\_tbids);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @" bool ProduceMeta(Dictionary<string, string> \_tbids)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

return false;

}

}

#endregion

#region 查询具有开发人员角色的用户

public DataSet GetDevelopMentUser()

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.GetDevelopMentUser();

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"DataSet GetDevelopMentUser()", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw;

}

}

#endregion

#region 查询具有测试人员角色的用户

public DataSet GetTestUser()

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.GetTestUser();

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"DataSet GetTestUser()", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw;

}

}

#endregion

#region 项目经理审核管理界面

/// <summary>

/// 项目经理返回草稿

/// </summary>

/// <param name="listTable"></param>

/// <param name="lstAuditDetail"></param>

/// <returns></returns>

public int BackToDraft(List<TblMetaDeTableMasterEntity> listTable, List<TblMetaDeAuditDetailEntity> lstAuditDetail)

{

try

{

IAuditInformationDAL \_AuditInformationDAL = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_AuditInformationDAL.BackToDraft(listTable, lstAuditDetail);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"BackToDraft(List<TblMetaDeTableMasterEntity> listTable, List<TblMetaDeAuditDetailEntity> lstAuditDetail)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

return 0;

}

}

/// <summary>

/// 项目经理返回草稿

/// </summary>

/// <param name="listTable"></param>

/// <param name="lstAuditDetail"></param>

/// <returns></returns>

public int SubmitMaintenance(List<TblMetaDeTableMasterEntity> listTable, List<TblMetaDeAuditDetailEntity> lstAuditDetail)

{

try

{

IAuditInformationDAL \_AuditInformationDAL = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_AuditInformationDAL.SubmitMaintenance(listTable, lstAuditDetail);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"SubmitMaintenance(List<TblMetaDeTableMasterEntity> listTable, List<TblMetaDeAuditDetailEntity> lstAuditDetail)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

return 0;

}

}

#endregion

#region 分配开发人员

/// <summary>

/// 分配开发人员

/// </summary>

/// <param name="tableEntitys"></param>

/// <returns></returns>

public bool AssignETLDevelopment(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> lstAuditDetail)

{

try

{

IAuditInformationDAL \_tablepromotedal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_tablepromotedal.AssignETLDevelopment(tableEntitys, lstAuditDetail);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @" bool ProduceMeta(Dictionary<string, string> \_tbids)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

return false;

}

}

#endregion

#region ETL开发人员操作界面

/// <summary>

/// 开发人员查询表信息

/// </summary>

/// <param name="tableEntitys"></param>

/// <returns></returns>

public PagingResult QueryDevelopmentTables(AuditQueryParam para)

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.QueryDevelopmentTables(para);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"PagingResult QueryDevelopmentTables(AuditQueryParam para)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw;

}

}

/// <summary>

/// 操作记录查询

/// </summary>

/// <param name="tbid"></param>

/// <returns></returns>

public DataSet QueryOperationList(string tbid)

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.QueryOperationList(tbid);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"DataSet QueryOperationList(string tbid)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw;

}

}

/// <summary>

/// 开发人员不通过

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public OperationResult DevelopmentReject(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.DevelopmentReject(tableEntitys, auditDetails);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"OperationResult DevelopmentReject(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw ex;

}

}

#endregion

#region 查询表审核信息

public PagingResult QueryTableAuditInfo(TableQueryParam QueryParam)

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.QueryTableAuditInfo(QueryParam);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"PagingResult QueryTableAuditInfo(TableQueryParam QueryParam)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw;

}

}

#endregion

#region 运维人员操作界面

/// <summary>

/// 试运行不通过

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public OperationResult TestRunReject(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.TestRunReject(tableEntitys, auditDetails);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"OperationResult TestRunReject(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw ex;

}

}

/// <summary>

/// 更新表状态为运行中

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public OperationResult UpdateTableStateToRunning(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.UpdateTableStateToRunning(tableEntitys, auditDetails);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"OperationResult UpdateTableStateToRunning(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw ex;

}

}

#region 生成元数据到正式库

/// <summary>

/// 生成元数据到正式库

/// </summary>

/// <param name="\_tbid"></param>

/// <returns></returns>

public OperationResult ProduceMetaToRun(List<GenerateMetaDataEntity> tables)

{

try

{

OperationResult oResult = new OperationResult();

IAuditInformationDAL \_tablepromotedal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_tablepromotedal.ProduceMetaToRun(tables);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @" OperationResult ProduceMetaToRun(List<GenerateMetaDataEntity> tables)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw ex;

}

}

#endregion

#region 生成元数据到试运行

/// <summary>

/// 生成元数据到试运行

/// </summary>

/// <param name="\_tbid"></param>

/// <returns></returns>

public OperationResult ProduceMetaToTestRun(List<GenerateMetaDataEntity> tables)

{

try

{

OperationResult oResult = new OperationResult();

IAuditInformationDAL \_tablepromotedal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

oResult= \_tablepromotedal.ProduceMetaToTestRun(tables);

if (oResult.IsSuccessed)

{

bool bResult= \_tablepromotedal.UpateTableIsGenerateData(tables, 2);

if(!bResult)

{

oResult.IsSuccessed = false;

oResult.Message = "元数据生成成功，但设计库表状态更新失败！";

}

}

return oResult;

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @" OperationResult ProduceMetaToRun(List<GenerateMetaDataEntity> tables)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw ex;

}

}

#endregion

#endregion

#region 测试任务管理界面

/// <summary>

/// 测试通过

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public OperationResult TestOK(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.TestOK(tableEntitys, auditDetails);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"OperationResult TestOK(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw ex;

}

}

/// <summary>

/// 测试不通过

/// </summary>

/// <param name="tableEntitys"></param>

/// <param name="auditDetails"></param>

/// <returns></returns>

public OperationResult TestReject(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)

{

try

{

IAuditInformationDAL \_auditinfodal = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_auditinfodal.TestReject(tableEntitys, auditDetails);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"OperationResult TestReject(List<TblMetaDeTableMasterEntity> tableEntitys, List<TblMetaDeAuditDetailEntity> auditDetails)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

throw ex;

}

}

#endregion

#region 开发人员提交测试

/// <summary>

/// 开发人员提交测试

/// </summary>

/// <param name="listTable"></param>

/// <param name="lstAuditDetail"></param>

/// <returns></returns>

public int SubmitTest(List<TblMetaDeTableMasterEntity> listTable, List<TblMetaDeAuditDetailEntity> lstAuditDetail)

{

try

{

IAuditInformationDAL \_AuditInformationDAL = MetaFactory.Factory.MetaDataAccess.CreateAuditInformationDAL();

return \_AuditInformationDAL.SubmitTest(listTable, lstAuditDetail);

}

catch (Exception ex)

{

BDDLogBLL.WebTxtErrorWirte("METADATA-DataBaseAction", @"SubmitTest(List<TblMetaDeTableMasterEntity> listTable, List<TblMetaDeAuditDetailEntity> lstAuditDetail)", ex.ToString(), "错误时间：" + DateTime.Now.ToString(), "暂无");

return 0;

}

}

#endregion

}

}