Account:

```
namespace Services.DAL.Account
{
        public class AccountOperator
        {
private static readonly string connectionString = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;



/// <summary>
///  用户注册
/// </summary>
/// <param name="model">注册对象</param>
/// <returns></returns>
public static ReturnState Register(RegisterView model)
{
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("insert into UserSets(Email, Pwd, uName, rDate, Tel, University) values (N'{0}', N'{1}', N'{2}', '{3}', N'{4}', {5})", model.Email, model.Password, model.UserName, DateTime.Now, model.Tel, model.University);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var result = cmd.ExecuteNonQuery();
        conn.Close();

        if (result <= 0)
        {
return ReturnState.ReturnError;
        }
}
        }
        return ReturnState.ReturnOK;
}

/// <summary>
///  查询邮箱是否存在
/// </summary>
/// <param name="emal"></param>
/// <returns></returns>
public static bool HasMember(string emal)
{
        bool result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from UserSets where email = '{0}'", emal);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteScalar() != null;
        conn.Close();
}
        }

        return result;
}

public static bool Login(LoginView model)
{
        bool result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from UserSets where email = '{0}' and pwd = '{1}'",
model.Email, model.Password);
using (var cmd = new SqlCommand(cmdText, conn))
```

```
{
        result = cmd.ExecuteScalar() != null;
        conn.Close();
}
        }

        return result;
}

public static UserInfoView GetUserInfo(string email)
{
        var user = new UserInfoView();
        user.Email = email;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select uName, Tel, c.name, Sex, Nick, Hobby, Avatar from UserSets a left join ExtraUserInfo b on a.Email = b.Email left join cfg_Universities c on a.university = c.Id where a.Email = N'{0}'", email);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        if (reader.HasRows)
        {
reader.Read();
user.UserName = Convert.ToString(reader.GetValue(0));
user.Tel = Convert.ToString(reader.GetValue(1));
user.University = Convert.ToString(reader.GetValue(2));
user.Sex = Convert.ToString(reader.GetValue(3));
user.Nick = Convert.ToString(reader.GetValue(4));
user.Hobby = Convert.ToString(reader.GetValue(5));
user.Avatar = Convert.ToString(reader.GetValue(6));
        }
        conn.Close();
}
        }

        return user;
}

public static bool UpdateUserInfo(UserInfoView model)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("update UserSets set uName = N'{0}', Tel = N'{1}' where Email = N'{2}'", model.UserName, model.Tel, model.Email);
var cmdText1 = string.Format(@"if not exists (select * from ExtraUserInfo where Email = N'{0}') insert into ExtraUserInfo values (N'{0}', N'{1}', N'{2}', N'{3}', N'{4}') else update ExtraUserInfo set Sex = N'{1}', Nick = N'{2}', Hobby = N'{3}', Avatar = N'{4}' where Email = N'{0}'", model.Email, model.Sex, model.Nick, model.Hobby, model.Avatar);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        cmd.CommandText = cmdText1;
        result = result && (cmd.ExecuteNonQuery() > 0);
        conn.Close();
}
        }

        return result;
}

public static bool UpdateUserPassword(string email, string password)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("update UserSets set Pwd = N'{1}' where Email = N'{0}'", email,
password);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }
        return result;
}

public static bool AdminLogin(LoginView model)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
}

Course:
namespace Services.DAL.Course
{
        public static class CourseOperator
        {
private static readonly string connectionString =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;



/// <summary>
/// 用户注册
/// </summary>
/// <param name="model">注册对象</param>
/// <returns></returns>
public static ReturnState AddCourse(CourseView model)
```

```
{
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("insert into CourseSets(Id, university, name, desp, pic_url) values (N'{0}',{1} , N'{2}', N'{3}', N'{4}')", model.Code, model.University, model.Name, model.Desp, model.PicUrl);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var result = cmd.ExecuteNonQuery();
        conn.Close();

        if (result <= 0)
        {
return ReturnState.ReturnError;
        }
}
        }
        return ReturnState.ReturnOK;
}

public static bool DeleteCourse(string code)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("delete from CourseSets where Id = N'{0}'", code);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }
        return result;
}

/// <summary>
///  查询课程编号是否存在
/// </summary>
/// <param name="emal"></param>
/// <returns></returns>
public static bool HasMember(string code)
{
        bool result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from CourseSets where Id = N'{0}'", code);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteScalar() != null;
        conn.Close();
}
        }

        return result;
}

public static string GetMaxCode()
{
        var result = "";
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select max(Id) from CourseSets ");
using (var cmd = new SqlCommand(cmdText, conn))
```

```
{
      result = Convert.ToString(cmd.ExecuteScalar());
      conn.Close();
}
      }

      return result;
}

/// <summary>
/// 分页查询
/// </summary>
/// <param name="page"></param>
/// <param name="nPage"></param>
/// <returns></returns>
public static List<CourseView> GetCourseByPage(int page, int nPage = 10)
{
      var retList = new List<CourseView>();
      using (var conn = new SqlConnection(connectionString))
      {
conn.Open();
var cmdText = string.Format("select top {0} * from CourseSets where Id not in (select top {1} Id from CourseSets)", nPage, page * nPage);
using (var cmd = new SqlCommand(cmdText, conn))
{
      var reader = cmd.ExecuteReader();
      while (reader.Read())
      {
var course = new CourseView()
{
      Code = Convert.ToString(reader.GetValue(0)),
      University = Convert.ToInt32(reader.GetValue(1)),
      Name = Convert.ToString(reader.GetValue(2)),
      Desp = Convert.ToString(reader.GetValue(3)),
        PicUrl = Convert.ToString(reader.GetValue(4))
};
retList.Add(course);
        }
}

return retList;
        }
}

public static CourseView GetCourseByCode(string code)
{
        CourseView ret = null;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from CourseSets where Id = N'{0}'", code);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var course = new CourseView()
{
        Code = Convert.ToString(reader.GetValue(0)),
        University = Convert.ToInt32(reader.GetValue(1)),
        Name = Convert.ToString(reader.GetValue(2)),
        Desp = Convert.ToString(reader.GetValue(3)),
        PicUrl = Convert.ToString(reader.GetValue(4))
};
ret = course;
        }
}
```

```
return ret;
        }
}

public static bool AddCourseApply(CourseView model, UserApply user)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("insert into tmp_CourseSets values (N'{0}', '{1}', '{2}', {3}, {4}, N'{5}', N'{6}', N'{7}')", model.Code, user.Email, DateTime.Now, user.Status, model.University, model.Name, model.Desp, model.PicUrl);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }

        return result;
}

public static List<CourseReviewView> GetCourseReviewViewByEmail(string email)
{
        var retList = new List<CourseReviewView>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select a.CommitDate, a.Code,a.name, a.desp, a.pic_url, b.Desp, c.name from tmp_CourseSets a left join cfg_ReviewStatus b on a.ReviewStatus = b.Id left join cfg_Universities c on a.university = c.Id where a.CommitUser = N'{0}'", email);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var course = new CourseReviewView()
{
        Email = email,
        CommitDate = Convert.ToDateTime(reader.GetValue(0)),
        Code = Convert.ToString(reader.GetValue(1)),
        Name = Convert.ToString(reader.GetValue(2)),
        Desp = Convert.ToString(reader.GetValue(3)),
        PicUrl = Convert.ToString(reader.GetValue(4)),
        Status = Convert.ToString(reader.GetValue(5)),
        University = Convert.ToString(reader.GetValue(6))
};
retList.Add(course);
        }
}
        }

        return retList;
}

public static List<CourseReviewView> GetCourseReviewViews()
{
        var retList = new List<CourseReviewView>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select a.CommitDate, a.Code,a.name, a.desp, a.pic_url, b.Desp, c.name, a.CommitUser from tmp_CourseSets a left join cfg_ReviewStatus b on a.ReviewStatus = b.Id left join cfg_Universities c on a.university = c.Id where a.ReviewStatus = 1");
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var course = new CourseReviewView()
{
        CommitDate = Convert.ToDateTime(reader.GetValue(0)),
        Code = Convert.ToString(reader.GetValue(1)),
        Name = Convert.ToString(reader.GetValue(2)),
        Desp = Convert.ToString(reader.GetValue(3)),
        PicUrl = Convert.ToString(reader.GetValue(4)),
        Status = Convert.ToString(reader.GetValue(5)),
        University = Convert.ToString(reader.GetValue(6)),
        Email = Convert.ToString(reader.GetValue(7))
};
retList.Add(course);
        }
}
        }

        return retList;
}
}
```

Market:

```
namespace Services.DAL.Market
{
        public class MarketOperator
        {
private static readonly string connectionString = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;

public static string NameOfSaleStatus(int id)
{
        var result = string.Empty;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Desp from cfg_SaleStatus where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var dbRet = cmd.ExecuteScalar();
        result = DBNull.Value.Equals(dbRet) ? string.Empty : Convert.ToString(dbRet);
        conn.Close();
}
        }
        return result;
}

public static int IndexOfSaleStatus(string name)
{
        var result = 0;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Id from cfg_SaleStatus where Desp = N'{0}'", name);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var dbRet = cmd.ExecuteScalar();
        result = DBNull.Value.Equals(dbRet) ? 1 : Convert.ToInt32(dbRet);
        conn.Close();
}
        }
        return result;
}

public static string NameOfGoodsType(int id)
{
        var result = string.Empty;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Desp from cfg_GoodsType where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var dbRet = cmd.ExecuteScalar();
        result = DBNull.Value.Equals(dbRet) ? string.Empty : Convert.ToString(dbRet);
        conn.Close();
}
        }
        return result;
}

public static int IndexOfGoodsType(string name)
{
        var result = 0;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Id from cfg_GoodsType where Desp = N'{0}'", name);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var dbRet = cmd.ExecuteScalar();
        result = DBNull.Value.Equals(dbRet) ? 1 : Convert.ToInt32(dbRet);
        conn.Close();
}
        }
        return result;
}

public static bool UserAddGoods(GoodsInfo model)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
```

```
    {
conn.Open();
var cmdText = string.Format("insert into GoodsSets values (N'{0}', N'{1}', {2}, N'{3}', N'{4}', '{5}', {6}, N'{7}', N'{8}', {9})", model.Seller, model.Name, model.Money, model.PicUrl, model.Desp, model.AddDate, IndexOfSaleStatus(model.Status), model.Buyer, model.Comments, IndexOfGoodsType(model.Type));
using (var cmd = new SqlCommand(cmdText, conn))
{
    result = cmd.ExecuteNonQuery() > 0;
    conn.Close();
}
    }

    return result;
}

public static bool UpdateGoodsInfoCommentById(int id, string content)
{
    var result = false;
    using (var conn = new SqlConnection(connectionString))
    {
conn.Open();
var cmdText = string.Format("update GoodsSets set Comment = N'{1}' where id = {0}", id, content);
using (var cmd = new SqlCommand(cmdText, conn))
{
    result = cmd.ExecuteNonQuery() > 0;
    conn.Close();
}
    }

    return result;
}

public static GoodsInfo SqlReaderGoodsInfo(SqlDataReader reader)
```

```
{
    var model = new GoodsInfo();
    model.Id = Convert.ToInt32(reader.GetValue(0));
    model.Seller = Convert.ToString(reader.GetValue(1));
    model.Name = Convert.ToString(reader.GetValue(2));
    model.Money = Convert.ToInt32(reader.GetValue(3));
    model.PicUrl = Convert.ToString(reader.GetValue(4));
    model.Desp = Convert.ToString(reader.GetValue(5));
    model.AddDate = Convert.ToDateTime(reader.GetValue(6));
    model.Status = NameOfSaleStatus(Convert.ToInt32(reader.GetValue(7)));
    model.Buyer = Convert.ToString(reader.GetValue(8));
    model.Comments = Convert.ToString(reader.GetValue(9));
    model.Type = NameOfGoodsType(Convert.ToInt32(reader.GetValue(10)));

    return model;
}

public static List<GoodsInfo> QueryGoodsInfoListByNameAndDesp(string filter, string findStr)
{
    var result = new List<GoodsInfo>();
    using (var conn = new SqlConnection(connectionString))
    {
conn.Open();
var cmdText = string.Format("select * from GoodsSets where {0} like N'%{1}%' and SStatus = 4",
filter, findStr);
using (var cmd = new SqlCommand(cmdText, conn))
{
    var reader = cmd.ExecuteReader();
    while (reader.Read())
    {
var model = SqlReaderGoodsInfo(reader);
result.Add(model);
    }
    conn.Close();
}
        }

        return result;
}

public static GoodsInfo QueryGoodsInfoById(int id)
{
        GoodsInfo result = null;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from GoodsSets where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        if (reader.Read())
        {
result = SqlReaderGoodsInfo(reader);
        }
        conn.Close();
}
        }

        return result;
}

public static List<GoodsInfo> GetGoodsInfoListBySeller(string seller)
{
        var result = new List<GoodsInfo>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from GoodsSets where Seller = N'{0}'", seller);
```

```
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var model = SqlReaderGoodsInfo(reader);
result.Add(model);
        }
        conn.Close();
}
        }

        return result;
}

public static List<GoodsInfo> GetGoodsInfoListByBuyer(string buyer)
{
        var result = new List<GoodsInfo>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from GoodsSets where Buyer = N'{0}'", buyer);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var model = SqlReaderGoodsInfo(reader);
result.Add(model);
        }
        conn.Close();
}
        }
        return result;
}

public static List<GoodsInfo> GetGoodsInfoListBySaleStatus(int status)
{
        var result = new List<GoodsInfo>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from GoodsSets where SStatus = {0}", status);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var model = SqlReaderGoodsInfo(reader);
result.Add(model);
        }
        conn.Close();
}
        }

        return result;
}

public static bool SetGoodsInfoStatusById(int id, int status)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("update GoodsSets set SStatus = {1} where Id = {0}", id, status);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }

        return result;
}

public static List<string> GetGoodsInfoTypeList()
{
        var result = new List<string>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Desp from cfg_GoodsType");
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while(reader.Read())
        {
result.Add(Convert.ToString(reader.GetValue(0)));
        }
}
        }
        return result;
}

public static List<GoodsInfo> GetAllGoodsInfo()
{
        var result = new List<GoodsInfo>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from GoodsSets");
```

```
using (var cmd = new SqlCommand(cmdText, conn))
{
    var reader = cmd.ExecuteReader();
    while (reader.Read())
    {
var model = SqlReaderGoodsInfo(reader);
result.Add(model);
    }
    conn.Close();
}
    }

    return result;
}

public static bool SetGoodsInfoSaleStatusAndBuyerById(int id, int status, string buyer)
{
    var result = false;
    using (var conn = new SqlConnection(connectionString))
    {
conn.Open();
var cmdText = string.Format("update GoodsSets set SStatus = {1}, buyer = N'{2}' where Id = {0}",
id, status, buyer);
using (var cmd = new SqlCommand(cmdText, conn))
{
    result = cmd.ExecuteNonQuery() > 0;
    conn.Close();
}
    }

    return result;
}

public static bool RemoveGoodsInfoById(int id)
```

```
{
    var result = false;
    using (var conn = new SqlConnection(connectionString))
    {
conn.Open();
var cmdText = string.Format("delete from GoodsSets where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
    result = cmd.ExecuteNonQuery() > 0;
    conn.Close();
}
    }

    return result;
}

public static bool AddLeaveMsg(LeaveMsgModel model)
{
    var result = false;
    using (var conn = new SqlConnection(connectionString))
    {
conn.Open();
var cmdText = string.Format("insert into LeaveMsg values({0}, N'{1}', '{2}', N'{3}', {4})",
model.Gid, model.Email, DateTime.Now, model.Content, GetValidFloorByGid(model.Gid));
using (var cmd = new SqlCommand(cmdText, conn))
{
    result = cmd.ExecuteNonQuery() > 0;
    conn.Close();
}
    }

    return result;
}
```

```
public static bool RemoveLeaveMsgById(int id)
{
    var result = false;
    using (var conn = new SqlConnection(connectionString))
    {
conn.Open();
var cmdText = string.Format("delete from LeaveMsg where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
    result = cmd.ExecuteNonQuery() > 0;
    conn.Close();
}
    }

    return result;
}

public static LeaveMsgModel SqlReaderLeaveMsg(SqlDataReader reader)
{
    var model = new LeaveMsgModel();
    model.Id = Convert.ToInt32(reader.GetValue(0));
    model.Gid = Convert.ToInt32(reader.GetValue(1));
    model.Email = Convert.ToString(reader.GetValue(2));
    model.PubDate = Convert.ToDateTime(reader.GetValue(3));
    model.Content = Convert.ToString(reader.GetValue(4));
    model.Floor = Convert.ToInt32(reader.GetValue(5));
    return model;
}

public static List<LeaveMsgModel> QueryLeaveMsgListByGid(int gid)
{
    var result = new List<LeaveMsgModel>();
    using (var conn = new SqlConnection(connectionString))
    {
```

```
conn.Open();
var cmdText = string.Format("select * from LeaveMsg where Gid = {0}", gid);
using (var cmd = new SqlCommand(cmdText, conn))
{
    var reader = cmd.ExecuteReader();
    while (reader.Read())
    {
result.Add(SqlReaderLeaveMsg(reader));
    }
    conn.Close();
}
    }

    return result;
}

public static LeaveMsgModel QueryLeaveMsgById(int id)
{
    LeaveMsgModel result = null;
    using (var conn = new SqlConnection(connectionString))
    {
conn.Open();
var cmdText = string.Format("select * from LeaveMsg where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
    var reader = cmd.ExecuteReader();
    if (reader.HasRows)
    {
result = SqlReaderLeaveMsg(reader);
    }
    conn.Close();
}
    }
```

```
        return result;
}
        }
}

Teacher:
namespace Services.DAL.Teacher
{
        public static class TeacherOperator
        {
private static readonly string connectionString =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;



/// <summary>
/// 用户注册
/// </summary>
/// <param name="model">注册对象</param>
/// <returns></returns>
public static ReturnState AddTeacherInfo(TeacherInfoView model)
{
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("insert into TeacherSets values (N'{0}', N'{1}', '{2}', N'{3}', N'{4}', {5},
{6}, N'{7}')", model.Email, model.UserName, DateTime.Now, model.Sex, model.Tel,
model.University, model.JobTitle, model.Desp);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var result = cmd.ExecuteNonQuery();
        conn.Close();

        if (result <= 0)
        {
```

```
return ReturnState.ReturnError;
        }
}
        }

        return ReturnState.ReturnOK;
}

/// <summary>
///  查询邮箱是否存在
/// </summary>
/// <param name="emal"></param>
/// <returns></returns>
public static bool HasMember(string emal)
{
        bool result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select * from TeacherSets where email = N'{0}'", emal);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteScalar() != null;
        conn.Close();
}
        }

        return result;
}

public static TeacherInfoView GetTeacherInfo(string email)
{
        var user = new TeacherInfoView();
        user.Email = email;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select uName, rDate, Sex, Tel, University, jTitle, Desp from
TeacherSets where Email = N'{0}'", email);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        if (reader.HasRows)
        {
reader.Read();
user.UserName = Convert.ToString(reader.GetValue(0));
user.RegisteDate = Convert.ToDateTime(reader.GetValue(1));
user.Sex = Convert.ToString(reader.GetValue(2));
user.Tel = Convert.ToString(reader.GetValue(3));
user.University = Convert.ToInt32(reader.GetValue(4));
user.JobTitle = Convert.ToInt32(reader.GetValue(5));
user.Desp = Convert.ToString(reader.GetValue(6));
        }
        conn.Close();
}
        }

        return user;
}

public static List<TeacherInfoView> GetTeacherInfoList()
{
        var retList = new List<TeacherInfoView>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select uName, rDate, Sex, Tel, University, jTitle, Desp, Email from
TeacherSets");
```

```
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var user = new TeacherInfoView();
user.UserName = Convert.ToString(reader.GetValue(0));
user.RegisteDate = Convert.ToDateTime(reader.GetValue(1));
user.Sex = Convert.ToString(reader.GetValue(2));
user.Tel = Convert.ToString(reader.GetValue(3));
user.University = Convert.ToInt32(reader.GetValue(4));
user.JobTitle = Convert.ToInt32(reader.GetValue(5));
user.Desp = Convert.ToString(reader.GetValue(6));
user.Email = Convert.ToString(reader.GetValue(7));
retList.Add(user);
        }
        conn.Close();
}
        }

        return retList;
}

public static bool UpdateUserInfo(TeacherInfoView model)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("update TeacherSets set uName = N'{1}', rDate = '{2}', Sex = N'{3}',
Tel = N'{4}', University = {5}, jTitle = {6}, Desp = N'{7}'   where Email = N'{0}'", model.Email,
model.UserName, model.RegisteDate, model.Sex, model.Tel, model.University, model.JobTitle,
model.Desp);
using (var cmd = new SqlCommand(cmdText, conn))
```

```
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }

        return result;
}

public static bool DelelteTeacherInfo(string email)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("delete from TeacherSets where Email = N'{0}'", email);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }

        return result;
}

public static bool DeleteTeacherCommnetById(int id)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("delete from TeacherCommentSets where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
```

```
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }

        return result;
}

public static bool AddCourseComment(TeacherCommentView model)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("insert into TeacherCommentSets values (N'{0}', N'{1}' , '{2}', N'{3}',
{4})", model.Teacher, model.Email, DateTime.Now, model.Content, model.Floor);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }
        return result;
}

public static bool RemoveCourseComment(TeacherCommentView model)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("delete from TeacherCommentSets where T_Id = N'{0}' and Email =
N'{1}' and cDate = '{2}'", model.Teacher, model.Email, model.PubDate);
```

```
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }

        return result;
}

public static List<TeacherCommentView> GetCourseCommentListByEmail(string email)
{
        var ret = new List<TeacherCommentView>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select T_Id, Email, cDate, Content, CmtFloor, Id from TeacherCommentSets where T_Id = N'{0}' order by CmtFloor DESC", email);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var comment = new TeacherCommentView();
comment.Teacher = Convert.ToString(reader.GetValue(0));
comment.Email = Convert.ToString(reader.GetValue(1));
comment.PubDate = Convert.ToDateTime(reader.GetValue(2));
comment.Content = Convert.ToString(reader.GetValue(3));
comment.Floor = Convert.ToInt32(reader.GetValue(4));
comment.Id = Convert.ToInt32(reader.GetValue(5));
ret.Add(comment);
        }
}
        }
        return ret;
}

public static List<string> GetTeacherByCourse(string code)
{
        var result = new List<string>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("");
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
result.Add(Convert.ToString(reader.GetValue(0)));
        }
        conn.Close();
}
        }
        return result;
}

        }
}
```

Forum:

```
namespace Services.DAL.Forum
{
        public static class ForumOperator
        {
private static readonly string connectionString =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
```

```
public static bool AddPost(PostModel model)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("insert into PostMsg values(N'{0}', N'{1}', {2}, N'{3}', '{4}', {5})",
model.Poster, model.Title, IndexOfPostType(model.PostType), model.Content, DateTime.Now,
model.NoComments);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }
        return result;
}

public static bool RemovePost(int id)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("delete from PostMsg where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }

        return result;
}

public static List<PostModel> QueryPostList()
{
        var ret = new List<PostModel>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Id, Email, Title, Content, PType, PDate, NoComments from
PostMsg order by PDate DESC");
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var model = new PostModel()
{
        Id = Convert.ToInt32(reader.GetValue(0)),
        Poster = Convert.ToString(reader.GetValue(1)),
        Title = Convert.ToString(reader.GetValue(2)),
        Content = Convert.ToString(reader.GetValue(3)),
        PostType = NameOfPostType(Convert.ToInt32(reader.GetValue(4))),
        PublishDate = Convert.ToDateTime(reader.GetValue(5)),
        NoComments = Convert.ToInt32(reader.GetValue(6))
};
ret.Add(model);
        }
        conn.Close();
}
        }

        return ret;
}
```

```
public static List<PostModel> QueryPostListByType(int type)
{
        var ret = new List<PostModel>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Id, Email, Title, Content, PType, PDate, NoComments from
PostMsg where PType = {0} order by PDate DESC", type);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var model = new PostModel()
{
        Id = Convert.ToInt32(reader.GetValue(0)),
        Poster = Convert.ToString(reader.GetValue(1)),
        Title = Convert.ToString(reader.GetValue(2)),
        Content = Convert.ToString(reader.GetValue(3)),
        PostType = NameOfPostType(Convert.ToInt32(reader.GetValue(4))),
        PublishDate = Convert.ToDateTime(reader.GetValue(5)),
        NoComments = Convert.ToInt32(reader.GetValue(6))
};
ret.Add(model);
        }
        conn.Close();
}
        }

        return ret;
}

public static List<PostModel> QueryPostListByEamil(string email)
{
        var ret = new List<PostModel>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Id, Email, Title, Content, PType, PDate, NoComments from
PostMsg where Email = N'{0}' order by PDate DESC", email);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var model = new PostModel()
{
        Id = Convert.ToInt32(reader.GetValue(0)),
        Poster = Convert.ToString(reader.GetValue(1)),
        Title = Convert.ToString(reader.GetValue(2)),
        Content = Convert.ToString(reader.GetValue(3)),
        PostType = NameOfPostType(Convert.ToInt32(reader.GetValue(4))),
        PublishDate = Convert.ToDateTime(reader.GetValue(5)),
        NoComments = Convert.ToInt32(reader.GetValue(6))
};
ret.Add(model);
        }
        conn.Close();
}
        }

        return ret;
}

public static PostModel QueryPostById(int id)
{
        PostModel model = null;
        using (var conn = new SqlConnection(connectionString))
```

```
    {
conn.Open();
var cmdText = string.Format("select Id, Email, Title, Content, PType, PDate, NoComments from
PostMsg where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
    var reader = cmd.ExecuteReader();
    while (reader.Read())
    {
model = new PostModel()
{
    Id = Convert.ToInt32(reader.GetValue(0)),
    Poster = Convert.ToString(reader.GetValue(1)),
    Title = Convert.ToString(reader.GetValue(2)),
    Content = Convert.ToString(reader.GetValue(3)),
    PostType = NameOfPostType(Convert.ToInt32(reader.GetValue(4))),
    PublishDate = Convert.ToDateTime(reader.GetValue(5)),
    NoComments = Convert.ToInt32(reader.GetValue(6))
};
    }
    conn.Close();
}
    }

    return model;
}

public static bool UpdatePost(PostModel model)
{
    var result = false;
    using (var conn = new SqlConnection(connectionString))
    {
conn.Open();
var cmdText = string.Format("update PostMsg set Title = N'{1}', Content = N'{2}', PType = {3},
NoComments = {4} where Id = {0}", model.Id, model.Title, model.Content,
IndexOfPostType(model.PostType), model.NoComments);
using (var cmd = new SqlCommand(cmdText, conn))
{
    result = cmd.ExecuteNonQuery() > 0;
    conn.Close();
}
    }

    return result;
}

public static List<PostReplyModel> QueryPostReplyListByPostId(int id)
{
    var list = new List<PostReplyModel>();
    using (var conn = new SqlConnection(connectionString))
    {
conn.Open();
var cmdText = string.Format("select Id, Email, Reply, Content, RDate from PostReply where Reply
= {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
    var reader = cmd.ExecuteReader();
    while (reader.Read())
    {
var model = new PostReplyModel();
model.Id = Convert.ToInt32(reader.GetValue(0));
model.Responser = Convert.ToString(reader.GetValue(1));
model.ResponseTo = Convert.ToInt32(reader.GetValue(2));
model.Content = Convert.ToString(reader.GetValue(3));
model.ResponseDate = Convert.ToDateTime(reader.GetValue(4));
list.Add(model);
    }
    conn.Close();
```

```
}
        }
        return list;
}

public static bool AddResponseToPost(PostReplyModel model)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("insert into PostReply values(N'{0}', {1}, N'{2}', '{3}')", model.Responser, model.ResponseTo, model.Content, DateTime.Now);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }

        return result;
}

public static bool RemoveResponseToPostById(int id)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("delete from PostReply where Id = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
```

```
}
        }

        return result;
}

public static List<PostReplyModel> QueryReplyResponseListByPostId(int id)
{
        var list = new List<PostReplyModel>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Id, Email, Reply, Content, RDate from PostReplyMsg where
Reply = {0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
var model = new PostReplyModel();
model.Id = Convert.ToInt32(reader.GetValue(0));
model.Responser = Convert.ToString(reader.GetValue(1));
model.ResponseTo = Convert.ToInt32(reader.GetValue(2));
model.Content = Convert.ToString(reader.GetValue(3));
model.ResponseDate = Convert.ToDateTime(reader.GetValue(4));
list.Add(model);
        }
        conn.Close();
}
        }
        return list;
}

public static PostReplyModel QueryPostReplyById(int id)
```

```
{
        var model = new PostReplyModel();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Id, Email, Reply, Content, RDate from PostReply where Id =
{0}", id);
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
model.Id = Convert.ToInt32(reader.GetValue(0));
model.Responser = Convert.ToString(reader.GetValue(1));
model.ResponseTo = Convert.ToInt32(reader.GetValue(2));
model.Content = Convert.ToString(reader.GetValue(3));
model.ResponseDate = Convert.ToDateTime(reader.GetValue(4));
        }
        conn.Close();
}
        }
        return model;
}

public static List<string> QueryPostTypeList()
{
        var result = new List<string>();
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("select Desp from cfg_PostType");
using (var cmd = new SqlCommand(cmdText, conn))
{
        var reader = cmd.ExecuteReader();
        while (reader.Read())
        {
result.Add(Convert.ToString(reader.GetValue(0)));
        }
}
        }
        return result;
}

public static bool AddResponseToPostReply(PostReplyModel model)
{
        var result = false;
        using (var conn = new SqlConnection(connectionString))
        {
conn.Open();
var cmdText = string.Format("insert into PostReplyMsg values(N'{0}', {1}, N'{2}', '{3}')", model.Responser, model.ResponseTo, model.Content, DateTime.Now);
using (var cmd = new SqlCommand(cmdText, conn))
{
        result = cmd.ExecuteNonQuery() > 0;
        conn.Close();
}
        }

        return result;
}
}
```