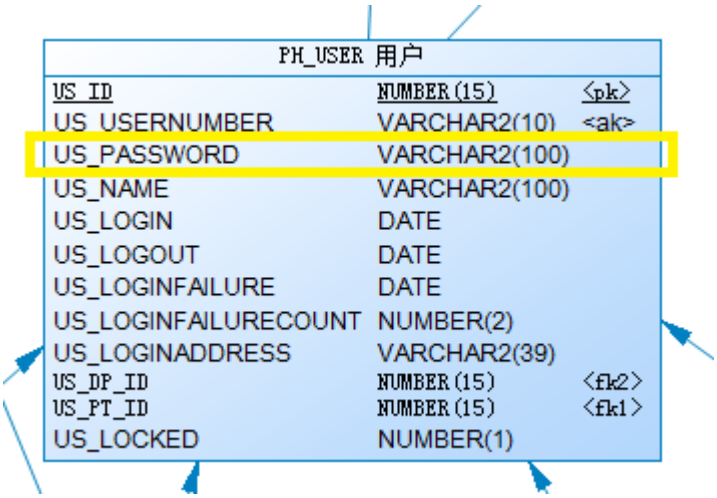
# 1 业务数据的读写-加解密

## 1.1 需求场景




类似 Phenix丶的 PH_User 表中的 US_Password 字段，保存的是密码之类的信息，要求即使直接检索数据库表记录，也不允许看到的是明文，这样的功能需求，需要我们在业务数据的读写（输入输出）过程中，将它做一次加解密的处理。

## 1.2 实现方法

Phenix丶在数据的加解密上封装了 System.Security.Cryptography.AESCryptoServiceProvider，由 Phenix.Core.Security.Cryptography.AesCryptoTextProvider 提供了如下简便的加解密函数：

```
/// <summary>
/// 加密
/// Key和IV将被转换为MD5值
/// </summary>
/// <param name="key">密钥</param>
/// <param name="IV">初始化向量</param>
/// <param name="value">需加密的字符串</param>
public static string Encrypt(string key, string IV, string value)

/// <summary>
/// 解密
/// Key和IV将被转换为MD5值
/// </summary>
/// <param name="key">密钥</param>
/// <param name="IV">初始化向量</param>
/// <param name="value">需解密的字节串</param>
public static string Decrypt(string key, string IV, string value)
```

这样，我们只要在属性的读写代码段中，插入加解密代码即可：

```
    private const string Key = "1234567890";
    private const string IV = "0987654321";

    /// <summary>
    /// 口令
    /// </summary>
    public static readonly Phenix.Business.PropertyInfo<string> PasswordProperty =
RegisterProperty<string>(c => c.Password, () => "TEMP");
    [Phenix.Core.Mapping.Field(FriendlyName = "口令", Alias = "US_PASSWORD", TableName = "PH_USER",
ColumnName = "US_PASSWORD", IsWatermarkColumn = true, NeedUpdate = true)]
    private string _password;
    /// <summary>
    /// 口令
    /// </summary>
    [System.ComponentModel.DisplayName("口令")]
    public string Password
    {
      get { return GetProperty(PasswordProperty, AesCryptoTextProvider.Decrypt(Key, IV, _password)); }
      set { SetProperty(PasswordProperty, ref _password, AesCryptoTextProvider.Encrypt(Key, IV,
value)); }
    }
```

注：本案例仅为示例。在实际业务场景下，Phenix丶已经将用户的口令做了加解密处理，而且即使在用户管理界面上，也是不应该让操作人员看到口令内容的，这个属性应该被注释掉。