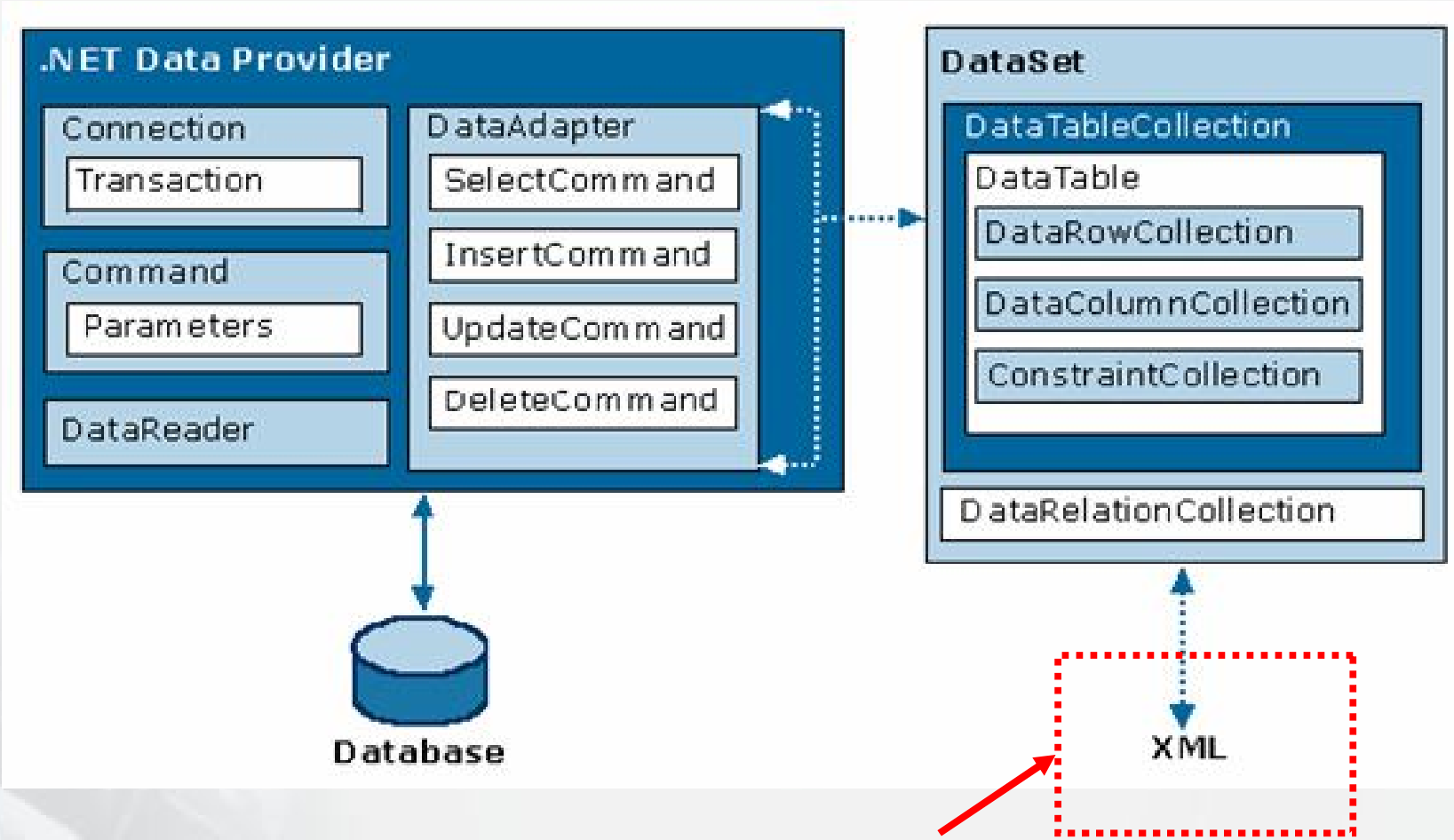
ĐỌC, GHI XML VỚI C#
TRONG ADO.NET
--- SỬ DỤNG VISUAL STUDIO 2010 ---

Nội dung

1. Tổng quan về XML
2. Đọc, ghi XML trong DataSet
3. Sử dụng DiffGrams trong XML


.NET Data Provider
Connection
Transaction
Command
Parameters
DataReader
DataAdapter
SelectCommand
InsertCommand
UpdateCommand
DeleteCommand
DataSet
DataTableCollection
DataTable
DataRowCollection
DataColumnCollection
ConstraintCollection
DataRelationCollection
Database
XML

1. Tổng quan về XML

- Khái niệm
- Tài liệu XML
- Các qui tắc khi tạo tài liệu XML

Khái niệm

- XML - E**X**tensible **M**arkup **L**anguage
- Dùng để lưu trữ và trao đổi dữ liệu
- Là một ngôn ngữ đánh dấu giống HTML
- Các Tag của XML không được định nghĩa trước, ta phải định nghĩa tag riêng ➔ tự mô tả
- Là một trong các chuẩn do W3C duy trì

Tài liệu XML

```
<?xml version="1.0" standalone="yes"?>
<DsChuDe>
  <ChuDe>
        <Mcd>3</Mcd>
        <TenChuDe>Tiếng Việt</TenChuDe>
  </ChuDe>
  <ChuDe>
        <Mcd>4</Mcd>
        <TenChuDe>Ngoại ngữ</TenChuDe>
  </ChuDe>
  <ChuDe>
        <Mcd>5</Mcd>
        <TenChuDe>Công nghệ thông tin</TenChuDe>
  </ChuDe>
  <ChuDe>
        <Mcd>7</Mcd>
        <TenChuDe>Luật</TenChuDe>
  </ChuDe>
</DsChuDe>
```

```
<?xml version="1.0" standalone="yes"?>
<DsChuDe>
  <ChuDe Mcd="3" TenChuDe="Tiếng Việt" />
  <ChuDe Mcd="4" TenChuDe="Ngoại ngữ" />
  <ChuDe Mcd="5" TenChuDe="Công nghệ thông tin" />
  <ChuDe Mcd="7" TenChuDe="Luật" />
</DsChuDe>
```

Các qui tắc tạo tài liệu XML

- Khi viết tài liệu XML ta phải tuân theo các qui tắc sau:
  - XML element phải có tag đóng
  - Tên tag phân biệt chữ HOA/thường
  - Các tag phải được lồng vào nhau có trình tự
  - Tài liệu XML chỉ có 1 element gốc
  - Giá trị của thuộc tính phải đặt trong dấu nháy đôi

Minh họa

- Tạo một tài liệu XML trong VS 2010

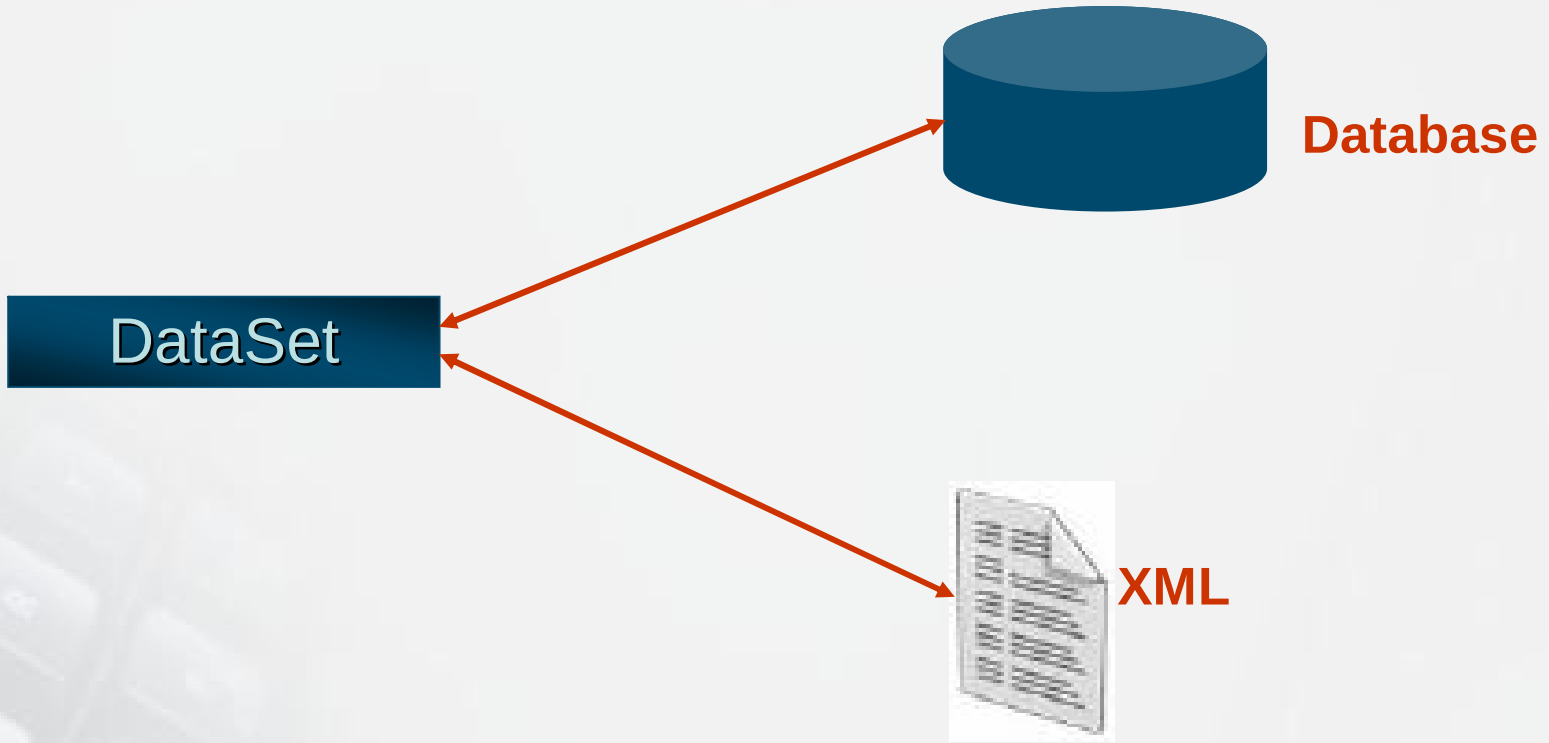
2. Đọc, ghi XML trong DataSet

- Tổng quan
- Ghi DataSet vào XML
- Đọc XML vào DataSet

Tổng quan

- ADO.NET cho phép đọc dữ liệu XML vào DataSet
- XML có thể chứa dữ liệu, lược đồ, hoặc cả hai.
- ADO.NET cũng cho phép ghi dữ liệu của DataSet vào XML với có hoặc không có cấu trúc
  - Gởi dữ liệu qua Internet cho một ứng dụng khác sử dụng

Tổng quan



Ghi DataSet vào XML

- Dùng 2 phương thức sau của DataSet
  - **WriteXml**(String [, XmlWriteMode])
    - XmlWriteMode: có ghi inline schema kèm theo dữ liệu không

Ghi DataSet vào XML

- Dùng 2 phương thức sau của DataSet

WriteSchema	Ghi nội dung hiện hành của Dataset có kèm theo inline XSD schema. Nếu Dataset chỉ có schema (không có dữ liệu) thì inline schema được ghi. Nếu Dataset không có schema hiện hành thì không ghi gì cả (cho dù có dữ liệu)
IgnoreSchema (Default)	Ghi nội dung hiện hành của Dataset không có XSD schema kèm theo. Nếu Dataset không có dữ liệu thì không ghi gì cả
DiffGram	Ghi toàn bộ Dataset có định dạng DiffGram, kể cả giá trị gốc và giá trị hiện hành

Ghi DataSet vào XML

- Dùng 2 phương thức sau của DataSet
  - **WriteXmlSchema**(String): ghi cấu trúc DataSet ra XML schema

Ví dụ: Đọc bảng Khoa từ CSDL QLSINHVIEN vào DataSet, sau đó ghi dữ liệu DataSet vào tập tin Khoa.Xml

```
//Đọc từ CSDL
SqlDataAdapter bo_doc_ghi = new SqlDataAdapter("SELECT * FROM Khoa", "Server=. ; Database=QLSINHVIEN;Integrated Security=SSPI");
DataSet ds = new DataSet();
bo_doc_ghi.Fill(ds);
//Ghi vào file XML
ds.WriteXml(Environment.CurrentDirectory + @"\Khoa.xml");
```

Minh họa lưu DataSet vào XML

- Đọc dữ liệu từ CSDL vào Dataset
- Ghi vào tập tin Xml dưới các chế độ khác nhau: thay đổi các hằng số của XmlWriteMode
- Mở các tập tin được tạo ra để xem kết quả

Đọc XML vào DataSet

- Dùng 2 phương thức sau của DataSet
  - **ReadXml**(String [, XmlReadMode])
    - XmlReadMode: xác định cách đọc dữ liệu XML và Schema liên quan
      - **Auto**: Gán vào XmlReadMode giá trị thích hợp nhất
        - » Nếu dữ liệu được định dạng là DiffGram, DiffGram được chọn.
        - » Nếu một inline schema được tìm thấy, thì **ReadSchema** được chọn.
        - » Nếu không có schema nào thì **IgnoreSchema** được chọn.

Đọc XML vào DataSet

- Dùng 2 phương thức sau của DataSet
  - **ReadXml**(String [, XmlReadMode])
    - **DiffGram**: Đọc trong DiffGram và áp dụng các thay đổi cho DataSet.
    - **IgnoreSchema:** Bỏ qua các inline schema. Đọc dữ liệu trong sơ đồ DataSet hiện tại. Nếu dữ liệu không tìm thấy trong DataSet schema nó được bỏ qua
    - **InferSchema:** Bỏ qua inline schema. Tạo schema dựa trên tài liệu XML. Nếu một schema có sẵn trong DataSet, schema này được sử dụng, và được mở rộng bằng cách thêm vào các cột và các bảng nếu cần. Có thể xảy ra một ngoại lệ nếu đã có một cột tồn tại sẵn, nhưng khác kiểu dữ liệu

Đọc XML vào DataSet

- Dùng 2 phương thức sau của DataSet
  - **ReadXml**(String [, XmlReadMode])
    - **ReadSchema**: Đọc một inline schema và load dữ liệu. Nếu Dataset đã có schema rồi thì các table mới sẽ được thêm vào schema. Nhưng nếu đã có table trong inline schema thì sẽ phát sinh một ngoại lệ
    - **InferTypedSchema**: Bỏ qua inline schema, định dạng dữ liệu dựa vào nội dung dữ liệu. Nếu không hiểu nội dung thì xem như là kiểu chuỗi

Đọc XML vào DataSet

- Dùng 2 phương thức sau của DataSet
  - **ReadXmlSchema**(String): đọc cấu trúc của DataSet mà không cần load dữ liệu

Ví dụ: Đọc tập tin XML chứa danh sách các khoa vào DataSet

```
DataSet ds = new DataSet();
ds.ReadXml(Environment.CurrentDirectory+ @"\khoa.xml",
XmlReadMode.Auto);
<DataGridView>.DataSource=ds.Tables[0];
```

Minh họa đọc XML vào DataSet

- Tạo Dataset
- Đọc nội dung các tập tin XML vào Dataset: thay đổi các hằng số XmlReadMode
- Xuất nội dung của Dataset ra màn hình
- Đọc tập tin chỉ lưu schema, sau đó xuất ra màn hình để xem tên cột và kiểu dữ liệu

3. Sử dụng DiffGrams trong XML

- DiffGrams là gì?
- Định dạng của DiffGram
- Các thuộc tính của DiffGram

DiffGrams là gì?

- Là một định dạng của XML
- Dùng để nhận dạng phiên bản gốc và hiện hành của các thành phần dữ liệu

Định dạng của DiffGram

- DiffGram gồm 3 khối thành phần như sau:
  - <DataInstance>, <diffgr:before>, <diffgr:errors>

```
<diffgr:diffgram
  xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
  xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

    <DataInstance> ... </DataInstance>
    <diffgr:before> ... </diffgr:before>
    <diffgr:errors> ... </diffgr:errors>

</diffgr:diffgram>
```

Định dạng của DiffGram

- DiffGram gồm 3 khối thành phần như sau:
  - Khối **<DataInstance>**
    - DataInstance chính là tên của thành phần này, thường là tên DataSet hoặc Datatable
    - Khối này dùng để chứa dữ liệu hiện hành
    - Khi dữ liệu thay đổi thì nó sẽ được nhận dạng thông qua thuộc tính diffgr:hasChanges

Định dạng của DiffGram

- DiffGram gồm 3 khối thành phần như sau:
  - **Khối <diffgr:before>**:
    - Khối này dùng để chứa phiên bản gốc của dữ liệu
    - Các thành phần trong khối này giống với khối <DataInstance> và được nhận dạng thông qua thuộc tính diffgr:id

Minh họa định dạng của DiffGram

- Tạo 2 Dataset riêng biệt
- Đọc cấu trúc của tập tin Xml schema và tập tin chứa nội dung có định dạng DiffGram vào Dataset 1
- Xuất nội dung Dataset 1 ra DataGridView 1
- Sửa dữ liệu trực tiếp trên DataGridView 1
- Ghi dữ liệu Dataset 1 ra 2 tập tin Xml khác (cấu trúc và nội dung có định dạng DiffGram)
- Đọc cấu trúc của tập tin Xml schema và tập tin chứa nội dung có định dạng DiffGram trên vào Dataset 2
- Xuất Dataset 2 ra DataGridView 2 để xem kết quả

Định dạng của DiffGram

- DiffGram gồm 3 khối thành phần như sau:
  - **Khối <diffgr:errors>**:
    - Khối này dùng để chứa thông tin lỗi của một dòng cụ thể trong khối <DataInstance>
    - Các thành phần trong khối này giống với khối <DataInstance> và được nhận dạng thông qua thuộc tính diffgr:id

Các thuộc tính của DiffGram

- Các thuộc tính đi kèm của các khối
  - **id**: định danh cho mỗi dòng, được kết hợp bởi [TableName][RowIdentifier]
  - **parentId**: nhận dạng thành phần cha của nó
  - **hasChanges**: nhận dạng thành phần hiện hành có được cập nhật không. Nếu có cập nhật thì dữ liệu gốc sẽ xuất hiện trong khối <diffgr:before>

Minh họa thuộc tính parentId

- Tạo 2 tập tin Xml: schema và nội dung có 2 table quan hệ cho con
- Xuất nội dung tập tin Xml ra 2 DataGridView Cha-Con
- Xóa và sửa dữ liệu trên DataGridView Con
- Ghi nội dung được cập nhật xuống tập tin Xml mới
- Mở tập tin mới này ra để xem nội dung thay đổi

Các thuộc tính của DiffGram

- Các thuộc tính đi kèm của các khối
  - **hasErrors**: nhận dạng thành phần hiện hành có bị lỗi hay không. Lúc đó dữ liệu gốc sẽ xuất hiện trong khối <diffgr:errors>
  - **Error**: chứa chuỗi mô tả lỗi trong khối <diffgr:errors>

Minh họa thuộc tính lỗi của DiffGram

- Tạo 1 Dataset và 1 DataAdapter
- Lắp dữ liệu từ bảng Sinh_vien vào Dataset
- Xuất nội dung Dataset ra DataGridView
- Sửa khoá ngoại – Ma_khoa – là một giá trị không tồn tại trên bảng cha
- Cập nhật Dataset và ghi ra Xml dưới dạng DiffGram
- Mở tập tin Xml này ra để xem nội dung

Minh họa một ứng dụng thực tế

- **Thi trắc nghiệm cuối khóa tại T3H**
  - Xuất đề thi trắc nghiệm (Ghi vào XML)
  - Tổ chức thi trắc nghiệm (Đọc, ghi XML)
  - Chấm thi (Đọc XML)

Ứng dụng .Net **"Quản lý kho đề thi trắc nghiệm"**

Xuất file trắc nghiệm

```
<?xml version="1.0"?>
<DeThi>
...
...
</DeThi>
```

Ứng dụng .Net "Thi trắc nghiệm"

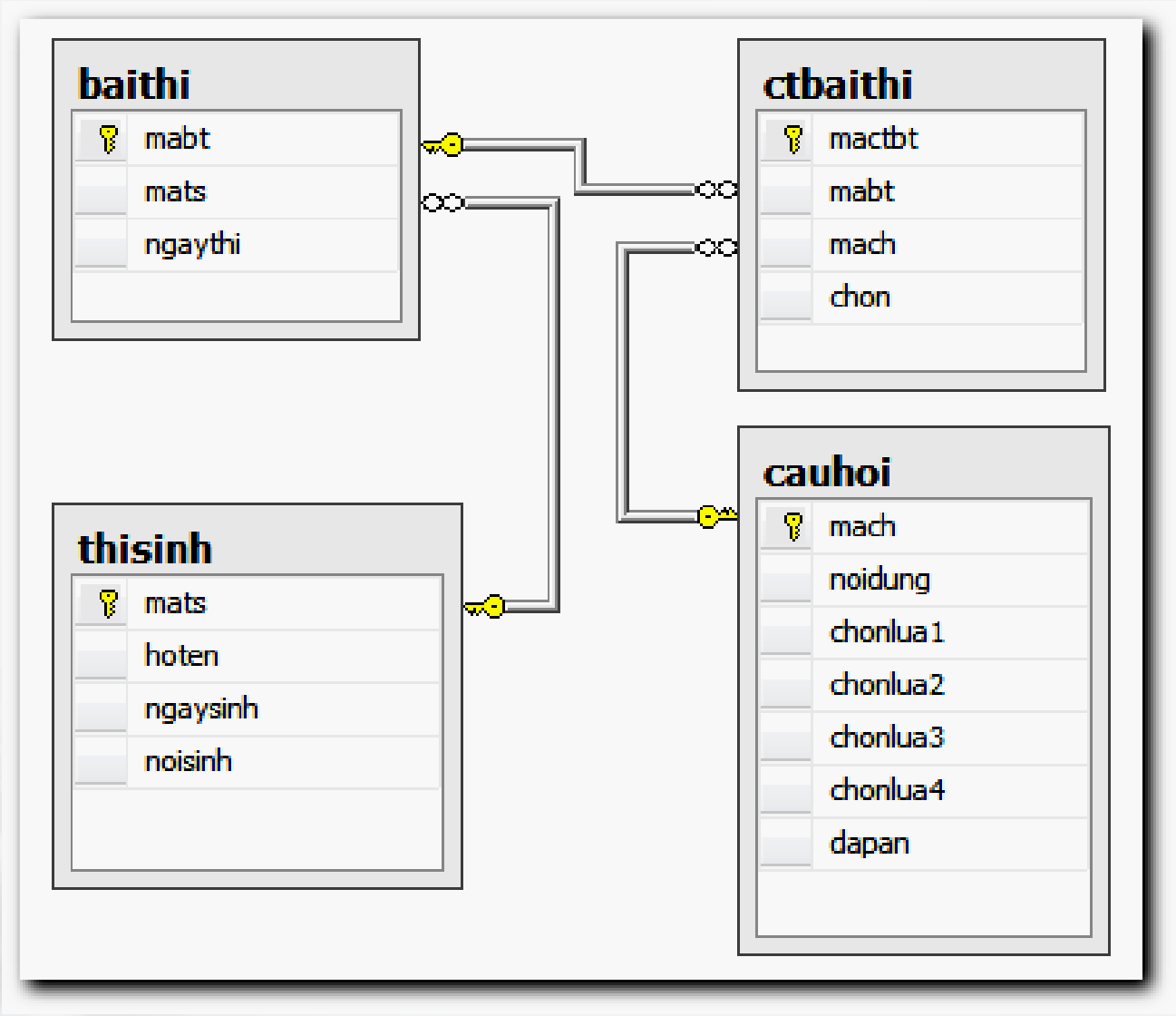
Tổ chức thi

Học viên nhập thông tin cá nhân và bắt đầu thi

Lưu bài thi

```
<?xml version="1.0"?>
<BaiThi>
...
...
</BaiThi>
```

Chấm thi


baithi
mabt
mats
ngaythi
ctbaithi
mactbt
mabt
mach
chon
cauhoi
mach
noidung
chonlua1
chonlua2
chonlua3
chonlua4
dapan
thisinh
mats
hoten
ngaysinh
noisinh

Hết