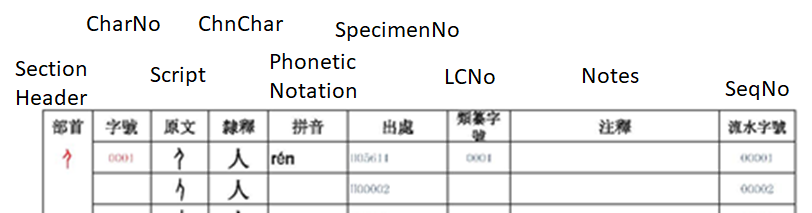
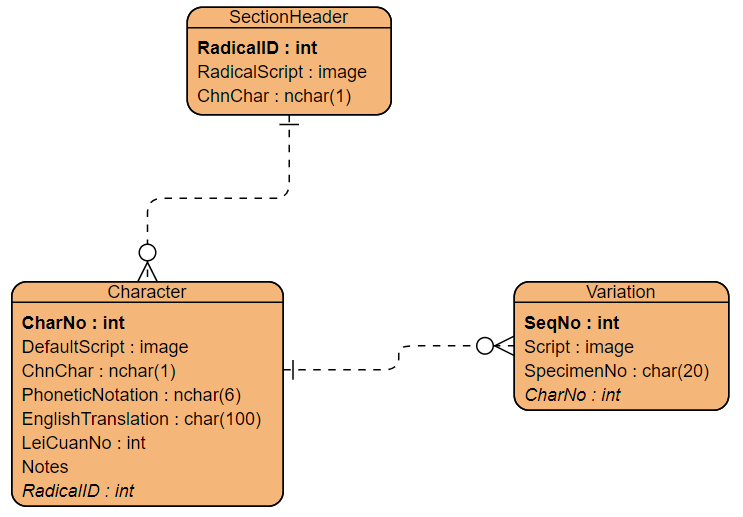
**Workshop 4 – Accessing Data on SQL Server through JDBC** (continued)

**Part 3 – Design (, Create,) and Use a New Database**

1. Use VP4UML to draw DB schema diagram, using the Entity Relationship Diagram. <https://online.visual-paradigm.com/app/diagrams/#diagram:proj=0&type=ERDiagram>
   1. The “table” (dataset) as shown on the sample page has the following attributes:  
         
      It can be normalized into three related “tables” (with a EngTrans field added):
      1. SectionHeader (RadicalID, Script, ChnChar),
      2. Character (CharNo, Script, ChnChar, PhoneticNotation, EnglishTranslation, LCNo, Notes),
      3. Variation (SeqNo, Script, SpecimentNo).



* 1. You may use VP online diagramming tools to draw a diagram as shown. It’s not easy to make changes, such as change attribute order in the list. You may use another tool, such as PPT, Visio, lucidchart, or edraw.

1. Read the CREATE TABLE statements that implement the required tables:

|  |
| --- |
| CREATE DATABASE OBScript;  -- go & use are TSQL commands  go  use OBScript  go  -- create tables  CREATE TABLE SectionHeader (  RadicalID int PRIMARY KEY,  Script varbinary(max),  ChnChar nchar(1)  );  CREATE TABLE Character (  CharNo int PRIMARY KEY,  DefaultScript varbinary(max),  ChnChar nchar(1),  PhoneticNotation nchar(6),  EnglishTranslation varchar(100),  LeiCuanNo int,  Notes nvarchar(100),  RadicalID int FOREIGN KEY  REFERENCES SectionHeader  );  CREATE TABLE Variation (  SeqNo int PRIMARY KEY,  Script varbinary(max),  SpecimentNo int,  CharNo int FOREIGN KEY  REFERENCES Character  );  -- nchar=native character type |

1. Insert data (numbers, images, Chinese and English character strings) into these tables
   1. General order: due to the dependency or FK referencing, records are added as below:
      1. First, a SectionHeader record,
      2. Then, (one or) more Character records,
      3. Lastly, (one or) more Variation records for each Character record
   2. Logics for parsing information from book pages:
      1. Save each page as an image file and locate script image location on page. (Ideally, we could divide book into pages and parse image from each page. But the scripts probably are stored as svg, which cannot be automatically retrieved by a standard API.)
      2. Clip areas containing script images and keep track numbers as ids
      3. Insert images and numbers into corresponding tables…