https://doi.org/10.48009/2_iis_2011_109-115

Issues in Information Systems

Volume XII, No. 2, pp 109-115, 2011

USING SQL DEVELOPER DATA MODELER AND MICROSOFT ACCESS FOR TEACHING DATABASE MODELING AND DESIGN

Kamal K. Hingorani, Alabama State University, khingorani@alasu.edu
Donald Woodard Alabama State University, dwoodard@alasu.edu
Narsin Askari-Danesh, Alabama State University, ndanesh@alasu.edu

ABSTRACT

Entity Relationship Diagrams (ERD) seems like a dying art, but many proponents believe that they should be the starting point of all software development projects. A new data modeling tool developed by Oracle could change the way data modeling in taught in today's classrooms. The powerful and easy to use tool that supports ERD allows a student to learn the entire cycle from logical to relational and finally to the physical design of the database. We present the way the tool was introduced in an introductory database management class. It is our belief that students can develop a higher level of expertise on the difficult concepts of database modeling and design with the aid of this tool.

Keywords: Information Technology (IT), Database, Relational Models, UML, ERD, Physical Models, and Logical models.

INTRODUCTION

With the growing popularity of Unified Modeling Language (UML), Entity Relationships Diagrams (ERD) seems to have lost some luster [3, 4, 7, 11]. Many of the textbooks on Introductory Database Management System and Systems Analysis and Design have reduced their coverage on the traditional ERD and now focus more on UML [7, 8]. Proponents of UML cite that class diagrams in UML are a superset of entity relationship modeling and there is nothing that can be expressed with ERDs that cannot be expressed in UML notation [9]. The traditionalists, however, respond that UML models an application, while ERD models reality. A relational database that contains the classes of an application will be highly application-specific while a relational database can be used by many more applications [1, 6, 12].

Oracle, which controls over 48% of the relational database management software, has recently introduced a stand-alone database modeling and design tool called SQL Developer Data Modeler (Version 3.0). This tool provides a full spectrum of data and database modeling tools and utilities, including modeling for Entity Relationship Diagrams (ERD), Relational (database design), Data Type and Multi-dimensional modeling, with forward and reverse engineering and DDL code generation.

This paper describes how SQL Developer Data Modeler was incorporated in an introductory database management course to teach students important concepts in database modeling and design. The DDL statements generated using the SQL Developer Data Modeler were used to create a database in Microsoft Access. The powerful visual features of Access helped reinforce the database design concepts. It is our belief that this database modeling tool makes a compelling case for continuing the teaching of ERD in the MIS curriculum. Students were able to correctly implement the entire cycle beginning from the logical modeling and ending with physical design with the help of this tool.

Oracle and MIS Programs

Oracle software and tools have not been extensively used in MIS programs. One reason is that Oracle, in the past, was a member of the anti-Windows parallel universe and concentrated on the UNIX platform. MIS programs relied on the Windows platform; the UNIX platform where Oracle concentrated on was the mainstay of Computer Science programs. With the growing popularity of the Windows Server platform, Oracle has developed strong tools and application that install and run seamlessly on Windows platform.

Oracle runs the Oracle Academy-an initiative that provides software, training, and teaching resources for free to Universities. Unfortunately, most of the success stories of this Academy are from schools (some of them high schools) in India and the East European countries. Our university is a member of all the three academies and has been teaching databases on the Oracle 11G platform for a number of years.

The Version 3.0 of the Oracle SQL Developer Data Modeler is a new tool that was released in January 2011. This tool is available for free and can be downloaded from http://www.oracle.com/technetwork/developer-tools/datamodeler/downloads/index.html . The installation is very simple; it requires unzipping the downloaded file and copying it to a folder on the hard-drive.

Oracle SQL Developer Data Modeler

Oracle, in its white paper, defines this product as a new, graphical data modeling tool that facilitates and enhances communication between data architects, database administrators, application developers and users, and simplifies the data modeling development process itself. Using SQL Developer Data Modeler users can create, browse and edit, logical, relational, physical, multi-dimensional, and data type models [10].

The generation of DDL scripts improves productivity and promotes the use of standards. SQL Developer Data Modeler model comprises three tightly

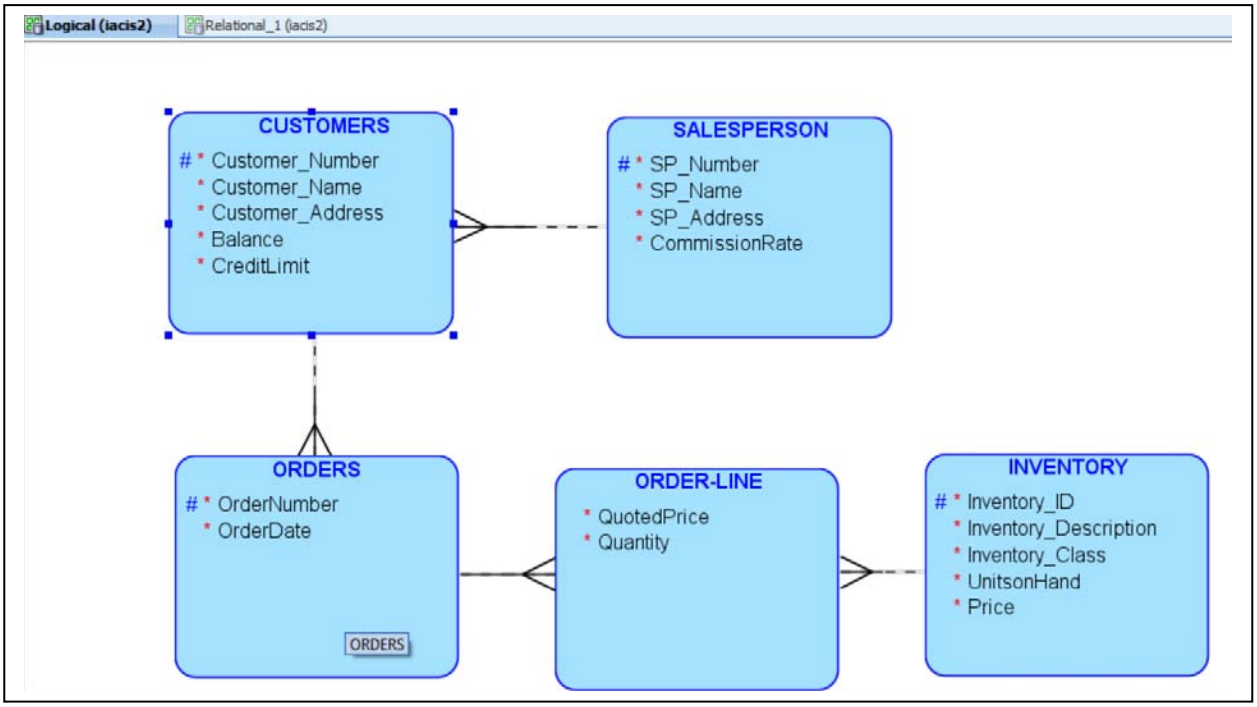

Figure 1: The Logical Model

synchronized layers: a logical model, relational models, and physical models. There is a one-to-many implementation when moving from the logical to the relational to the physical layer, which means that you can create multiple instances at the next, lower level in the hierarchy. In simple terms, the product is an enterprise-class modeling tool with a very compelling front-end.

The logical model in SQL Developer Data Modeler includes standard logical modeling facilities, such as drawing entities and relationships, plus the following key features:

- Box-in-box presentation for the super-type and sub-pety hierarchy of entities

- Support for exclusive relationships (arcs)
- Barker or Bachman notation (the Barker notations are the default) [2,5]

The relational model is an intermediate model between the logical model and the physical models. It supports relational design decisions independent of the constraints of the target physical platform(s). All many-to-many relationships and all supertype/sub-types entity hierarchies are resolved during forward engineering (transformation) of the logical model, or part of it, to a relational model.

The physical model supports Oracle 9i, Oracle 10g, Oracle 11g, SQL Server 2005, SQL Server 2000, DB2 Version 8, and DB2 Version 7 objects.

TEACHING METHODOLOGY

Our teaching centered upon developing a business scenario that would enable students to design a logical, relational, and a physical database model that would incorporate all the important aspects of the ERD.
The scenario's logical model included the following:

- Entity with attributes of different data types
- A weak entity
- Relationships
- Cardinality constraints
- Participation constraints

The concepts of Generalization/specialization and the super-type and sub-type hierarchy of entities were left out to reduce the complexity of the exercise.

The business scenario presented to the students was:

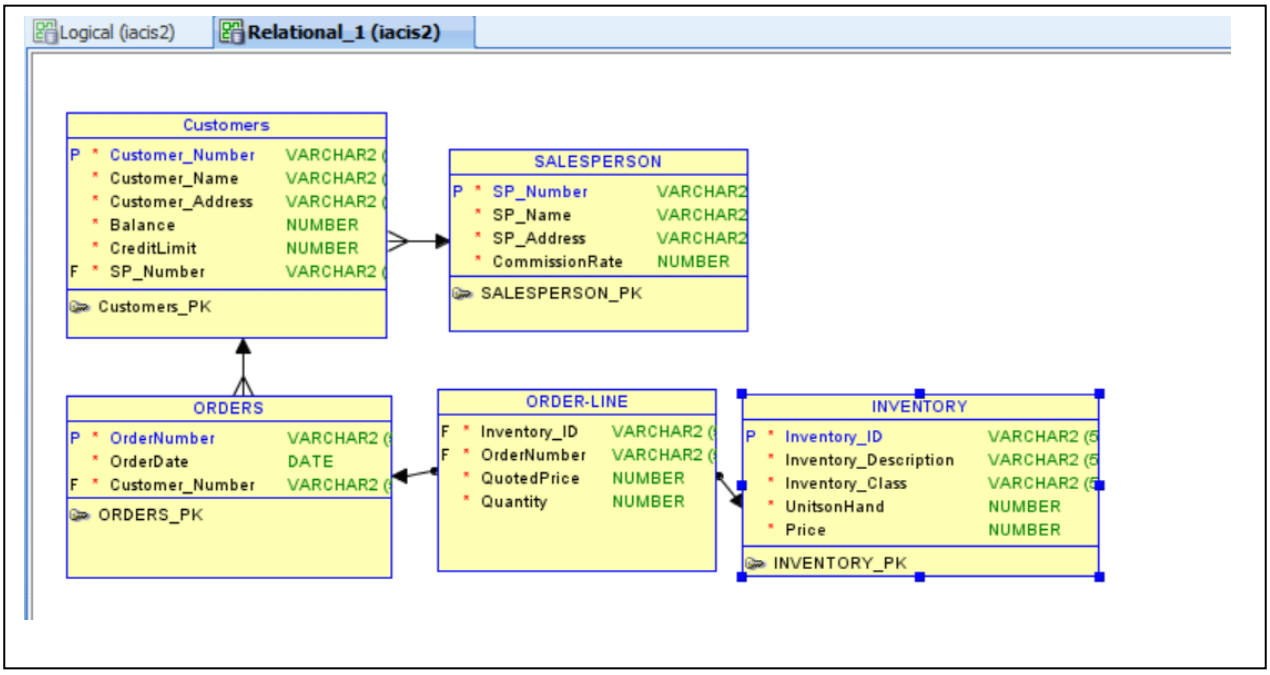

Figure 2: The Relational Model

We are a retail organization. We keep track of a salesperson's name, number, address, and commission rate. Customers are our bread and butter. Each customer opens an account with us and we keep track of their name,

number, address, balance, and credit limit. Each customer is assigned a unique salesperson. We keep track of our inventory by focusing on its number, description, item class, price, and unit on hand. A customer usually places many orders with us. We keep a track of the order number and the date. An order can contain multiple items. We usually quote a price for each item.

OUTCOMES

The resulting ER model (logical) as drawn in SQL Developer Data Modeler is shown in Figure 1. The scenario results in 5 entities and 4 relationships. The attributes (required or mandatory), key attribute, cardinality and participating constraints are all depicted in easy to understand model. While developing the model, the tool allows the user to specify the data types. Since the physical model was to be implemented in Access, the students were asked to use only the following data-types: varchar, date, and numeric (without the precision and the scale)

The relational model for the logical model can be generated through a process called forward engineering. This process requires just one click of the mouse. The relational model is shown in Figure-2
The relational model converted the data-type varchar to varchar2. VARCHAR2 is a data type in Oracle that is used to store variable-length character data.

The relational model allows the user to generate the DDL statements. Although the tool allows you a choice of seven models, the one that can be used in Access with the least amount of modification is Oracle 11g. The resulting physical model (DDL statements) is shown in Appendix-1. The physical model is made of five Create Table SQL statements for generating the tables and eight Alter table SQL statements for implementing the primary keys and the foreign keys. The DDL statements can then be used to create the database in Access, the varchar2 data-type has to be changed to varchar before the statements can be run in Access. Each of the thirteen (five Create Table and eight Alter Table) SQL statements have to be run separately as MS Access does not support batch SQL statements.
The SQL Developer Data Modeler also supports reverse engineering. MS Access, although a great tool for teaching database modeling and design, does not have the feature of creating a data dictionary in XML format. The database documenter does generate the object definition, but the format is a report that cannot be imported into the Data Modeler. SQL Developer Data Modeler is very well synchronized with the Oracle SQL Developer, which is a front end for working with Oracle Databases. The SQL Developer requires access to an Oracle server.

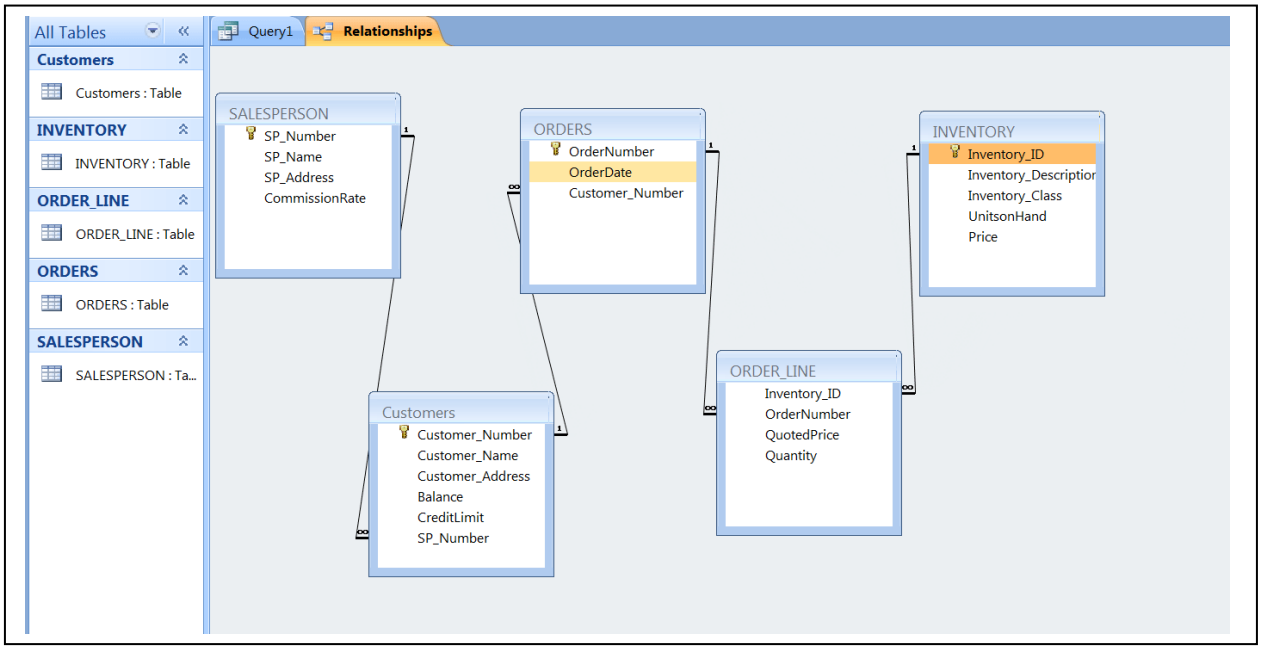

Figure 3: The Physical Database in MS Access

CONCLUSIONS

The aptitudes, attitudes, expectations, and learning styles of students (the Net Generation) puts pressure on universities to offer courses that incorporate the state-of-the-art technologies and are practical in nature. Employers now require our graduates to seamlessly transition to the workforce with little or no "on-the-job" training. Universities are therefore, actively seeking new teaching tools for the classroom.

Database modeling and design is an important skill set for new hires. Universities are always looking for tools that can be used for imparting the requisites skills. With so much emphasis on UML, the traditional ERD modeling tool has taken a back seat in most database courses in MIS programs across the nation. The SQL Developer Data Modeler, a robust data modeling tool developed by Oracle, is a tool that can make teaching and learning of database design and database programming easier and students can complete the introductory database courses with far greater level of expertise.

REFERENCES

1. Bock, D. B. and Yaeger, S. E. (2002). Improving Entity Relationship Modeling Accuracy with Novice Data Modelers. *Journal of Computer Information Systems 42*(2), 69-75.
2. Chen, P. P. (1976). The Entity-Relationship Model: Toward a Unified View of Data," *ACM Transactions on Database Systems, 1*(1), 9-36.
3. *Gravino, De Lucia, Oliveto* R. , and *Tortora G. (2010)* An experimental comparison of ER and UML class diagrams for data modeling Journal of Empirical Software Engineering ,Volume 15 Issue 5, October 2010
4. Grossman, M., McCarthy, R.V, and Aronson, J.E.,, Factors influencing Unified Modeling Language (UML) usage in the global software development community, *Fourth Annual Global Information Technology Management (GITM) World Conference, Calgary, Canada* (2003) pp. 294–297.
5. Hitchman, Steve (2002) "The Details of Conceptual Modeling Notations are Important - A Comparison of Relationship Normative Language," Communications of the Association for Information Systems: Vol. 9, Article 10.
6. Kobryn, C Will UML 2.0 be agile or awkward?, *Communications of the ACM* **45** (2002) (1), pp. 107–110
7. Mrdalj, S. & Jovanovic, V. (2004). UML Coverage in Systems Analysis and Design Textbooks. *Issues in Information Systems, 5*(1), 233-239.
8. Neubauer, Bruce Data modeling in the understanding database course: adding UML and XML modeling to the traditional course content, Journal of Computing Sciences in Colleges, v.17 n.5, p.147-153, April 2002
9. Object Management Group (2006). Unified Modeling Language (UML), version 2.0. http://www.omg.org/technology/documents/formal/ uml.htm
10. Oracle (2011) An Introduction to Oracle Developer Data Modeler, available at www.oracle.com
11. Pons, A. P, Polak, P. & Stutz, J. (2006). Evaluating the Teaching Effectiveness of Various Data Modeling Notations. *Journal of Computer Information Systems, 46*(2), 78-84.
12. Siau K. and Cao,Q. Unified Modeling Language (UML): a complexity analysis, *Journal of Database Management* **12** (2001) (1), p. 26.

APPENDIX A
DDL STATEMENTS FOR THE PHYSICAL MODEL

```
CREATE TABLE Customers
(Customer_Number VARCHAR2 (50)  NOT NULL ,
 Customer_Name VARCHAR2 (50)  NOT NULL ,
 Customer_Address VARCHAR2 (50)  NOT NULL ,
   Balance NUMBER  NOT NULL ,
   CreditLimit NUMBER  NOT NULL ,
   SP_Number VARCHAR2 (50)  NOT NULL) ;

ALTER TABLE Customers ADD CONSTRAINT Customers_PK PRIMARY KEY (Customer_Number) ;

CREATE TABLE INVENTORY
  (Inventory_ID VARCHAR2 (50)  NOT NULL ,
   Inventory_Description VARCHAR2 (50)  NOT NULL ,
   Inventory_Class VARCHAR2 (50)  NOT NULL ,
   UnitsonHand NUMBER  NOT NULL ,
   Price NUMBER  NOT NULL) ;

ALTER TABLE INVENTORY
  ADD CONSTRAINT INVENTORY_PK PRIMARY KEY (Inventory_ID) ;
CREATE TABLE "ORDER-LINE"
  (Inventory_ID VARCHAR2 (50)  NOT NULL ,
   OrderNumber VARCHAR2 (50)  NOT NULL ,
   QuotedPrice NUMBER  NOT NULL ,
   Quantity NUMBER  NOT NULL) ;

CREATE TABLE ORDERS
  (OrderNumber VARCHAR2 (50)  NOT NULL ,
   OrderDate DATE  NOT NULL ,
   Customer_Number VARCHAR2 (50)  NOT NULL) ;

ALTER TABLE ORDERS
  ADD CONSTRAINT ORDERS_PK PRIMARY KEY (OrderNumber) ;

CREATE TABLE SALESPERSON
  (SP_Number VARCHAR2 (50)  NOT NULL ,
   SP_Name VARCHAR2 (50)  NOT NULL ,
   SP_Address VARCHAR2 (50)  NOT NULL ,
   CommissionRate NUMBER  NOT NULL) ;

ALTER TABLE SALESPERSON
  ADD CONSTRAINT SALESPERSON_PK PRIMARY KEY (SP_Number) ;

ALTER TABLE Customers
  ADD CONSTRAINT Relation_2 FOREIGN KEY
  (SP_Number)
  REFERENCES SALESPERSON
  (SP_Number) ;

ALTER TABLE ORDERS
  ADD CONSTRAINT Relation_3 FOREIGN KEY
  (Customer_Number)
```

```
  REFERENCES Customers
  (Customer_Number) ;

ALTER TABLE ORDER_LINE
  ADD CONSTRAINT Relation_5 FOREIGN KEY
  (OrderNumber)
  REFERENCES ORDERS
  (OrderNumber)
  ON DELETE CASCADE ;

ALTER TABLE ORDER_LINE
  ADD CONSTRAINT Relation_6 FOREIGN KEY
  (Inventory_ID)
  REFERENCES INVENTORY
  (Inventory_ID) ;
 (All varchar2 need to be converted to varchar for MS Access)
*
```