Data Modeling and Database Design Using ERwin
Main Features: ORACLE Designer Data Modeling Tool

ORACLE Designer 2000 Compatible

Main Features: Importing Entities from BPwin

BPWin is an activity (process) modeling, and data flow diagramming tool

Main Features: BPWin ve ERWin

System Development Life Cycle

Main Features: Working Levels

1. Logical (Conceptual) Level (Data Modeling)
2. Physical Level

Main Features: Working Levels (for Version 3.5.2)

1. Logical Level (Data Modeling)

Adding Entity
Identifying relationship
Non-identifying Relationship
many-to-many relationship

Main Features: Working Levels (For Version 4)

1. Logical Level (Data Modeling)

Adding Entity
Identifying relationship
Non-identifying Relationship
many-to-many relationship
Working Levels

2. Physical Level

(Version 3.5.2)

Independent Table

View

Adding Text

Identifying Relationship

Non-identifying Relationship

Selection

Changing Columns

Identifying Relationship

View Relationship

ERWIN and Its Main Features

Modeling and Design Stages for ERwin

PRODUCE DATA MODEL

DESCRIBE DATA TYPES

SETTINGS FOR ACCESSING DATABASE

VIA SQL SCRIPTS

BY CONNECTING TO DATABASE

REVERSE ENGINEERING

Logical Level

Physical Level

Native or ODBC

Forward Engineering

Representing Entities and Attributes (Logical Level)

ENTITY NAME

Unique Identifier Attributes

Other Attributes

Foreign Key Attribute

Composite Unique Identifiers

Entity Name

Composite Unique Identifier

Other Attributes
Optionality in Attributes

Click twice on any entity to open it in attribute editor

Representing Foreign Key Attributes

There are two different relationships between Entity 1 and Entity 2.
1. Identifying Relationships
2. Non-Identifying Relationships

In both relationships, Unique Identifier in master entity goes to the second (child) entity as foreign key.

Identifying Relationship

(Unique Identifier of Entity 1 becomes the part of unique identifier in entity 2. It is placed in key area of second entity)

Non-Identifying Relationship

(Unique Identifier of Entity 1 becomes new attribute in entity 2. It is placed in non-key area of second entity)

Use Different names for the attributes in key area (unique identifier)

Overwrites the attribute
Use Different names for the attributes in key area (unique identifier)

Tables and Columns (Physical Representation)

Table Name

**EMPLOYEES**
- **employee_id**: CHAR(18)
- **first_name**: CHAR(18)
- **surname**: CHAR(18)
- **date_of_birth**: CHAR(18)
- **country_of_origin**: CHAR(18)
- **manager_id**: CHAR(18)

Unique Identifier Column

**EMPLOYEES**
- **FIRST_NAME**: VARCHAR(20)
- **SURNAME**: VARCHAR(20)
- **DATE_OF_BIRTH**: DATE
- **COUNTRY_OF_ORIGIN**: VARCHAR(20)
- **MANAGER_ID**: INT

Other Columns

Note: All Attributes are assigned as char(18) (or any default value defined) when the level is changed to physical one.

Changing default data types for Physical level

Adding Icons to Entities and Attributes
Establishing Non-identifying Relationship

1. Choose Non-identifying Relationship
2. Click the master entity (DEPARTMENT in this case)
3. Click the child entity (EMPLOYEE)

DEPARTMENT ID in DEPARTMENT comes as foreign key to the non-key area of EMPLOYEE

Establishing Identifying Relationship

1. Choose Identifying Relationship
2. Click the master entity (EMPLOYEE in this case)
3. Click the child entity (EMPLOYEE CHILD)

EMPLOYEE ID in EMPLOYEE comes as foreign key to the key area of EMPLOYEE CHILD

Representing ORACLE Designer Notations under ERWIN

1. Many-to-one (One-to-many) Relationships (Mandatory)

Child Entity

Parent Entity

non-identifying Relationship

The entity covering the other entity is defined as child entity in one-to-one relationship.

Representing ORACLE Designer Notations under ERWIN

2. Many-to-one (One-to-many) Relationships (Optional)

Child Entity

Parent Entity

non-identifying Relationship

Representing ORACLE Designer Notations under ERWIN

3. One-to-one Relationships (Optional)

Child Entity

Parent Entity

non-identifying Relationship

Representing ORACLE Designer Notations under ERWIN

4. One-to-one Relationships (Mandatory)

Child Entity

Parent Entity

non-identifying Relationship

The entity covering the other entity is defined as child entity in one-to-one relationship.
5. Many-to-many Relationships

Note: First, you have to resolve many-to-many relationships

Intersection Entity Added Automatically

It becomes identifying relationship

6. Obtaining Uniqueness via Relationships

<table>
<thead>
<tr>
<th>ACCOUNT</th>
<th>8ª Account No</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BANK</td>
<td>8ª Bank ID</td>
<td>Name</td>
</tr>
</tbody>
</table>

Represented as identifying relationship

7. Recursive Relationships

Modeled as Non-Identifying Relationship as in one-to-many relationship

ERwin
1. First, identify the entity and its unique identifier
2. Choose non-identifying relationship from toolbox
3. Click twice on the same entity (EMPLOYEE)
4. EMPLOYEE ID has not come as Foreign Key
5. Click twice on the relationship to identify rolename
6. MANAGER ID comes as Foreign key.
8. Exclusive Relationships

Transferred to ERwin for each pair of relationships

9. Supertypes and Subtypes

Modeled as Identifying Relationship

Choosing Target Database Management System

Choosing Database

Database/Choose Database for selecting target database management system

NOTE: You have to be under level

Transforming Data Model To Database Design
Supported Database Management Systems

1. IBM DB2
2. IBM AS/400
3. ORACLE
4. MS SQL Server
5. InterBase
6. SYBASE
7. PROGRESS
8. INFORMIX
9. Ingres
10. Red Brick
11. SQLBase
12. Rdb
13. WATCOM/SQL Anywhere
14. Teradata
15. Clipper
16. FoxPro
17. Access
18. Paradox
19. SQLBase
20. Rdb
21. SQL Anywhere
22. Teradata
23. Clipper
24. FoxPro
25. Access
26. Paradox

You can also access to the DBMSs not listed here via ODBC/Generic

Transforming Data Model To Database Design

Choosing Target Database Management System

Making Connection Settings for Target Database Management System

ODBC Settings For connecting to DBMS

Before Transforming Data Model

For adding new connection

For connecting to Oracle DBMS

Example; Settings for connecting to Oracle DBMS through ODBC

Connection name to connect
(You can reach under ERWin ODBC Connection)

Service name

Username

Example; MS SQL Server Connections

Connection Name for SQL Server

Optional Description

ODBC Settings for each DBMS is different (like printer drivers)
Forward Engineering

Applying the principles of Systems Development Life Cycle (SDLC)
(Going from the top to bottom in Waterfall model)

Producing the Relational Database Objects via the use of Data Models

Choose Forward Engineer Under Tools

Transforming Data Model To Database Design

Forward Engineering: Settings

Options

Filtering model
Preview objects
Print SQL Script
Write SQL Script
to a File
Generate
Database
Objects

ODBC Connection
Native Connection

Press Generate button

1. Press
2. Give required security information and press Connect button.

Produce the database objects from data model using SQL Script

Press button

SQL Script filename

Generate SQL Server/SQL Schema Report

Save as type: SQL Script
Save
**Transforming Data Model To Database Design**

Produce the database objects from data model using SQL Script

```
CREATE TABLE EMPLOYEE (EMPLOYEE_ID NUMBER NOT NULL,
                         FIRST_NAME VARCHAR2(30) NOT NULL,
                         LAST_NAME VARCHAR2(30) NOT NULL,
                         ADDRESS VARCHAR2(100),
                         EMAIL VARCHAR2(50),
                         PHONE NUMBER(12),
                         JOB VARCHAR2(30),
                         HIRE_DATE DATE NOT NULL);

CREATE TABLE DEPARTMENT (DEPARTMENT_ID NUMBER NOT NULL,
                          DEPARTMENT_NAME VARCHAR2(30),
                          MANAGER_ID NUMBER,
                          LOCATION VARCHAR2(30));

CREATE TABLE DEPARTMENT_HOURS (DEPARTMENT_ID NUMBER,
                               HOURS NUMBER(4));

CREATE TABLE JOB_HISTORY (EMPLOYEE_ID NUMBER,
                          FROM_DATE DATE NOT NULL,
                          TO_DATE DATE,
                          JOB_ID NUMBER,
                          DEPARTMENT_ID NUMBER);

CREATE TABLE PROJECT (PROJECT_ID NUMBER NOT NULL,
                       PROJECT_NAME VARCHAR2(30),
                       START_DATE DATE NOT NULL,
                       END_DATE DATE);
```

Just run the SQL Script File using tools under the DBMSs

**Examples**

- Oracle SQL*Plus for Oracle DBMS
- Query Analyzer for MS SQL Server
- Interactive SQL for Interbase DBMS

**Reverse Engineering**

Applying the principles of Systems Development Life Cycle (SDLC) in reverse order
(Going from the bottom to top in Waterfall model)

Inferring the Data Model from Database Objects

**Select Reverse Engineer under Tools Menu**
Reverse Engineering: Settings

As in forward engineering, connections are achieved via the native or ODBC type of connection

Reverse Engineering

Choose the Target DBMS to reverse engineer the database

Reverse Engineering

Data model can be obtained from:
1. DBMS by direct connection
2. SQL Script Files

If the database is not relational then you can infer primary keys and relationships from indexes and the column names in tables

Reengineering

Other Functions

Reengineering: Compares the data model and physical database. You can update the data model via the database or database via the data model

ERwin Reports

Report Generation

For report generation
ERwin Reports

Reports

Reports: Reports are generated Using Report Templates and Output type (such as HTML)

A typical data model using ERwin

Sample data model using Oracle Designer
Sample data model using ERwin