ММинистерство образования и науки Российской Федерации
Федеральное агентство по образованию
КНИТУ - КАИ им. А. Н. Туполева
Институт Компьютерных технологий и Защиты информации
Кафедра Прикладной математики и информатики
им. Ю. В. Кожевникова

Лабораторная работа №4

«Создание модели данных с помощью Allfusion ERwin Data Modeler»

по дисциплине «Проектирование и архитектура программных систем»

Выполнил
студент группы 4312
Наумихин Д.Д.
Нарыжный И.Д.

Казань, 2025

Цель работы

Знакомство с CASE-системой Allfusion ERWin Data Modeler, изучение основных принципов построения логической модели данных, разработка модели.

Отображение модели данных в ERwin

Физическая и логическая модель данных

ERwin имеет два уровня представления модели - логический и физический.

Логический уровень - это абстрактный взгляд на данные, на нем данные представляются так, как выглядят в реальном мире, и могут называться так, как они называются в реальном мире, например "Постоянный клиент", "Отдел" или "Фамилия сотрудника". Объекты модели, представляемые на логическом уровне, называются сущностями и атрибутами (подробнее о сущностях и атрибутах будет рассказано ниже). Логическая модель данных может быть построена на основе другой логической модели, например, на основе модели процессов. Логическая модель данных является универсальной и никак не связана с конкретной реализацией СУБД.

Физическая модель данных, напротив, зависит от конкретной СУБД, фактически являясь отображением системного каталога. В физической модели содержится информация о всех объектах БД. Поскольку стандартов на объекты БД не существует (например, нет стандарта на типы данных), физическая модель зависит от конкретной реализации СУБД. Следовательно, одной и той же логической модели могут соответствовать несколько разных физических моделей. Если в логической модели не имеет значения, какой конкретно тип данных имеет атрибут, то в физической модели важно описать всю информацию о конкретных физических объектах - таблицах, колонках, индексах, процедурах и т. д. Разделение модели данных на логические и физические позволяет решить несколько важных задач.

Документирование модели. Многие СУБД имеют ограничение на именование объектов (например, ограничение на длину имени таблицы или запрет использования специальных символов - пробела и т. п.). Зачастую разработчики ИС имеют дело с нелокализованными версиями СУБД. Это означает, что объекты БД могут называться короткими словами, только латинскими символами и без использования специальных символов (т. е. нельзя назвать таблицу предложением - только одним словом). Разделение модели на логическую и физическую позволяет решить эту проблему. На физическом уровне объекты БД могут называться так, как того требуют ограничения СУБД. На логическом уровне можно этим объектам дать синонимы - имена более понятные неспециалистам, в том числе на кириллице и с использованием специальных символов. Такое соответствие позволяет лучше задокументировать модель и дает возможность обсуждать структуру данных с экспертами предметной области.

Масштабирование. Создание модели данных, как правило, начинается с создания логической модели. После описания логической модели, проектировщик может выбрать необходимую СУБД и ERwin автоматически создаст соответствующую физическую модель. На основе физической модели ERwin может сгенерировать системный каталог СУБД или соответствующий SQL-скрипт. Этот процесс называется прямым проектированием (Forward Engineering). Тем самым достигается масштабируемость - создав одну логическую модель данных, можно сгенерировать физические модели под любую поддерживаемую ERwin СУБД. С другой стороны, ERwin способен по содержимому системного каталога или SQL-скрипту воссоздать физическую и логическую модель данных (Reverse Engineering). На основе полученной логической модели данных можно сгенерировать физическую модель для другой СУБД и затем сгенерировать ее системный каталог. Следовательно, ERwin позволяет решить задачу по переносу структуры данных с одного сервера на другой.

Например, можно перенести структуру данных с Oracle на Informix (или наоборот).

Для переключения между логической и физической моделью данных служит список выбора в левой части панели инструментов Erwin (Рис.).

Рис.1. Переключение между логической и физической моделью

При переключении, если физической модели еще не существует, она будет создана автоматически.

Подмножества модели и сохраняемые отображения

При создании реальных моделей данных количество сущностей и атрибутов может исчисляться сотнями. Для более удобной работы с большими моделями в ERwin предусмотрены подмножества модели (Subject Area), в которые можно включить тематически общие сущности. В подмножество модели может входить произвольный набор сущностей, связей и текстовых комментариев. Для создания, удаления или редактирования подмножеств модели нужно вызвать диалог Subject Area Editor (меню Edit/Subject Area), в котором указывается имя подмножества и входящие в нее сущности (Рис. 1). Все изменения, сделанные в любой Subject Area, автоматически отображаются на общей модели. Одна и та же сущность может входить в несколько Subject Area.

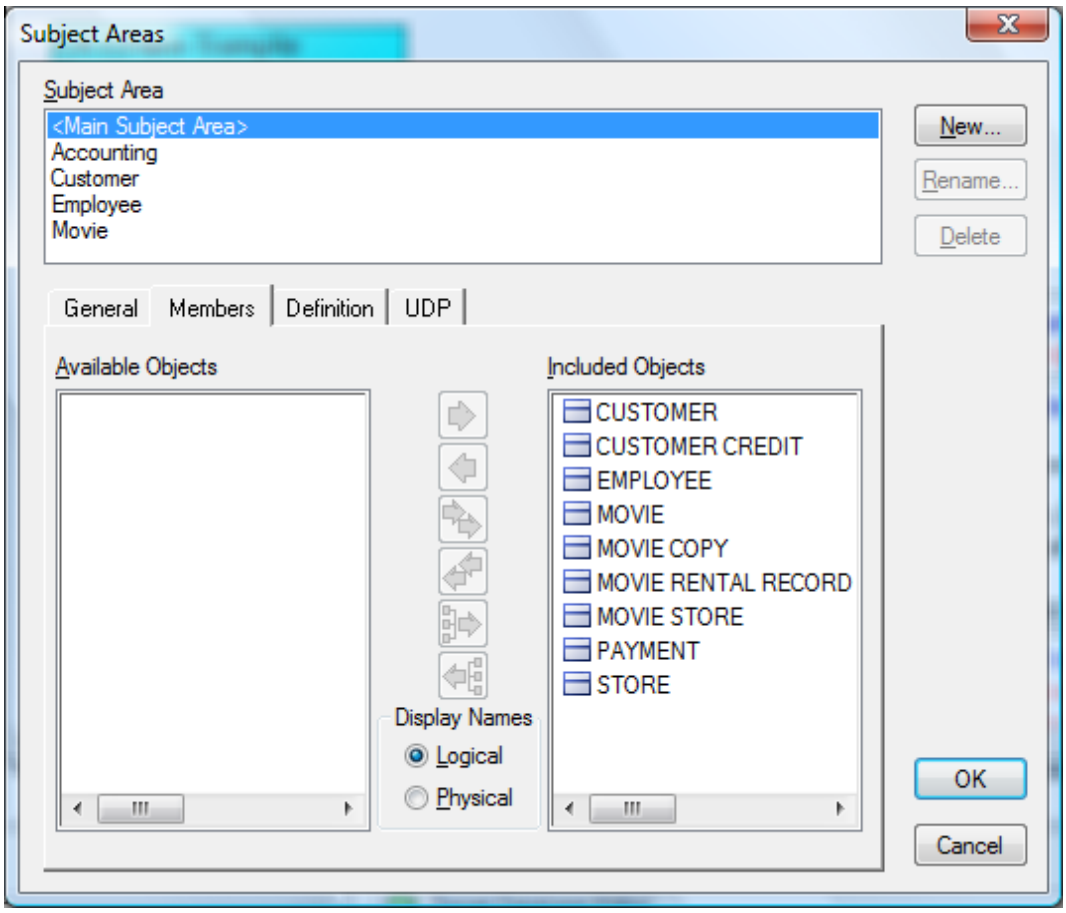

Рис. 1. Диалог Subject Area Editor

По умолчанию исходная модель получает имя Main Subject Area. При создании нового подмножества следует в диалоге Subject Area Editor указать ее имя и список входящих в него объектов. Subject Area можно создавать как в логической, так и в физической модели данных.

Создание логической модели данных

Уровни логической модели

Различают три уровня логической модели, отличающихся по глубине представления информации о данных:

диаграмма сущность-связь (Entity Relationship Diagram, ERD);

модель данных, основанная на ключах (Key Based model, KB);

полная атрибутивная модель (Fully Attributed model, FA).

Диаграмма сущность-связь представляет собой модель данных верхнего уровня. Она включает сущности и взаимосвязи, отражающие основные бизнес-правила предметной области. Такая диаграмма не слишком

детализирована, в нее включаются основные сущности и связи между ними, которые удовлетворяют основным требованиям, предъявляемым к ИС. Диаграмма сущность-связь может включать связи многие-ко-многим и не включать описание ключей. Как правило, ERD используется для презентаций и обсуждения структуры данных с экспертами предметной области.

Модель данных, основанная на ключах, - более подробное представление данных. Она включает описание всех сущностей и первичных ключей и предназначена для представления структуры данных и ключей, которые соответствуют предметной области.

Полная атрибутивная модель - наиболее детальное представление структуры данных: представляет данные в третьей нормальной форме и включает все сущности, атрибуты и связи.

Сущности и атрибуты

Основные компоненты диаграммы Erwin - это сущности, атрибуты и связи. Каждая сущность является множеством подобных индивидуальных объектов, называемых экземплярами. Каждый экземпляр индивидуален и должен отличаться от всех остальных экземпляров. Атрибут выражает определенное свойство объекта. С точки зрения БД (физическая модель) сущности соответствует таблица, экземпляру сущности - строка в таблице, а атрибуту - колонка таблицы.

Построение модели данных предполагает определение сущностей и атрибутов, т. е. необходимо определить, какая информация будет храниться в конкретной сущности или атрибуте. Сущность можно определить как объект, событие или концепцию, информация о которых должна сохраняться. Сущности должны иметь наименование с четким смысловым значением, именоваться существительным в единственном числе, не носить "технических" наименований и быть достаточно важными для того, чтобы их моделировать. Именование сущности в единственном числе облегчает в дальнейшем чтение модели. Фактически имя сущности дается по имени ее экземпляра. Примером может быть сущность *Заказчик* с атрибутами *Номер*

заказчика, Фамилия заказчика и Адрес заказчика. На уровне физической модели ей может соответствовать таблица *Customer с* колонками *Customer_number, Customer_name* и *Customer_address.*

Каждая сущность должна быть полностью определена с помощью текстового описания в закладке Definition. Закладки Note, Note 2, Note 3, UDP (User Defined Properties - Свойства, определенные пользователем) служат для внесения дополнительных комментариев и определений к сущности.

Закладка Definition используется для ввода определения сущности. Эти определения полезны как на логическом уровне, поскольку позволяют понять, что это за объект, так и на физическом уровне, поскольку их можно экспортировать как часть схемы и использовать в реальной БД (CREATE COMMENT on entity_name).

Закладка Note позволяет добавлять дополнительные замечания о сущности, которые не были отражены в определении, введенном в закладке Definition. Здесь можно ввести полезное замечание, описывающее какое-либо бизнес-правило или соглашение по организации диаграммы.

В закладке Note 2 можно задокументировать некоторые возможные запросы, которые, как ожидается, будут использоваться по отношению к сущности в БД. При переходе к физическому проектированию, записанные запросы помогут принимать такие решения в отношении проектирования, которые сделают БД более эффективной.

Связи

Связь является логическим соотношением между сущностями. Каждая связь должна именоваться глаголом или глагольной фразой (Verb Phrases) (Рис. 2). Имя связи выражает некоторое ограничение или бизнес-правило и облегчает чтение диаграммы.

.

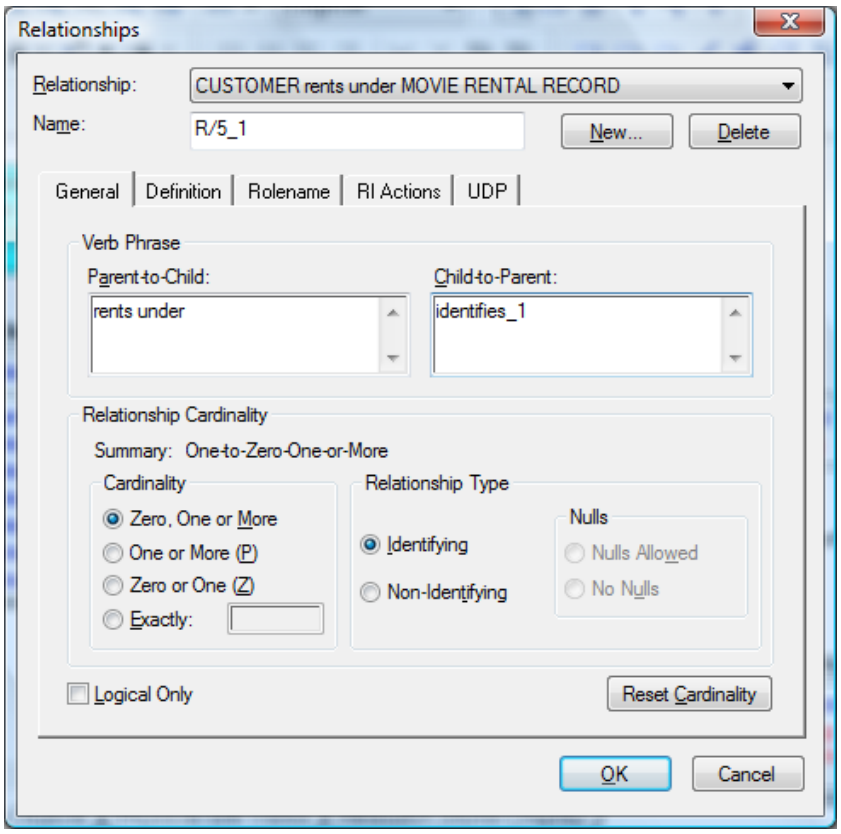

Рис. 2. Диалог Relationships

Связь показывает, какие именно заказы разместил клиент и какой именно сотрудник выполняет заказ. По умолчанию имя связи на диаграмме не показывается. Для отображения имени следует в контекстном меню, которое появляется, если щелкнуть левой кнопкой мыши по любому месту диаграммы, не занятому объектами модели, выбрать пункт Display Options/Relationship и затем включить опцию Verb Phrase.

На логическом уровне можно установить идентифицирующую связь один-ко-многим, связь многие-ко-многим и неидентифицирующую связь один-ко-многим (соответственно это кнопки слева направо в палитре инструментов).

В IDEF1X различают зависимые и независимые сущности. Тип сущности определяется ее связью с другими сущностями. Идентифицирующая связь устанавливается между независимой (родительский конец связи) и зависимой (дочерний конец связи) сущностями. Когда рисуется идентифицирующая связь, ERwin автоматически преобразует дочернюю сущность в зависимую. Зависимая

сущность изображается прямоугольником со скругленными углами. Экземпляр зависимой сущности определяется только через отношение к родительской сущности. При установлении идентифицирующей связи атрибуты первичного ключа родительской сущности автоматически переносятся в состав первичного ключа дочерней сущности. Эта операция дополнения атрибутов дочерней сущности при создании связи называется миграцией атрибутов. В дочерней сущности мигрировавшие атрибуты помечаются как внешний ключ - (FK).

В дальнейшем, при генерации схемы БД, атрибуты первичного ключа получат признак NOT NULL, что означает невозможность внесения записи в таблицу заказов без информации о номере клиента.

При установлении неидентифицирующей связи дочерняя сущность остается независимой, а атрибуты первичного ключа родительской сущности мигрируют в состав неключевых компонентов родительской сущности. Неидентифицирующая связь служит для связывания независимых сущностей.

Задание на лабораторную работу

Построить логическую модель данных, включающую основные сущности и связи, для разработки информационной системы в следующей предметной области: Автоматизированная информационная система учета аренды автомобилей.

Результат выполнения: