Лабораторная работа №10. Проектирование хранилищ данных.

ЦЕЛЬ РАБОТЫ: Познакомиться с понятием хранилища данных, его отличием от базы данных, основными принципами его организации, ознакомиться с методикой построения хранилищ данных в ERwin и применить полученные знания для разработки собственного хранилища данных.

Хранилища данных (Data Warehouse) представляют собой специализированные базы данных, предназначенные для хранения данных, которые редко меняются, но на основе которых часто требуется выполнение сложных запросов. Обычно они ориентированы на выполнение аналитических запросов, которые обеспечивают поддержку принятия решений для руководителей и менеджеров. Хранилища данных позволяют разгрузить промышленные базы данных, и тем самым, позволяют пользователям более эффективно и быстро извлекать необходимую информацию. Как правило, хранилища данных оперируют с огромными объемами информации, что предъявляет к их проектированию и реализации повышенные требования.

Выбор в качестве платформы хранилища данных такой высокопроизводительных РСУБД позволяет существенно повысить общую эффективность создаваемой информационной системы. Рассмотрим основные возможности ERwin по проектированию хранилищ данных.

К проектированию хранилищ данных обычно предъявляются следующие требования:

- Структура данных хранилища должна быть понятна пользователям;
- Должны быть выделены статические данные, которые регулярно модифицируются: ежедневно, еженедельно, ежеквартально;
- Должны быть упрощены требования к запросам с целью исключения запросов, которые могли бы требовать множественных утверждений SQL в традиционных реляционных СУБД;
- Должна быть обеспечена поддержка сложных запросов SQL, которые требуют последовательной обработки тысяч или миллионов записей.

Эти требования существенно отличают структуру реляционных СУБД и хранилищ данных. Нормализация данных в реляционных СУБД приводит к созданию множества связанных между собой таблиц. В результате, выполнение сложных запросов неизбежно приводит к объединению многих таблиц, что существенно увеличивает время отклика. Проектирование хранилища данных подразумевает создание денормализованной структуры

данных (допускается избыточность данных и возможность возникновения аномалий при манипулировании данными), ориентированной в первую очередь на высокую производительность при выполнении аналитических запросов.

Для эффективного проектирования хранилищ данных ERwin использует размерную (*Dimensional*) модель, специально предназначенная для разработки хранилищ данных. Наиболее простой способ перейти к нотации *Dimensional* - при создании новой модели (меню *File→New*) в диалоге *ERwin Teamplate Selection* выбрать из списка предлагаемых шаблонов *DIMENSION*.

Моделирование *Dimensional* сходно с моделированием связей и сущностей для реляционной модели, но отличаются, целями. Реляционная модель акцентируется на целостности и эффективности ввода данных. Размерная (*Dimensional*) модель ориентирована в первую очередь на выполнение сложных запросов к БД.

В размерном моделировании принят стандарт модели, называемый схемой звезда (*star schema*), которая обеспечивает высокую скорость выполнения запроса, посредством денормализации и разделения данных. Схема звезда строится так, чтобы обеспечить наивысшую производительность при выполнении одного самого важного запроса, либо для группы похожих запросов.

Схема звезда обычно содержит одну большую таблицу, называемую таблицей факта (*fact table*), помещенную в центр, и окружающие ее меньшие таблицы, называемые таблицами размерности (*dimensional table*), соединенными с таблицей факта в виде звезды радиальными связями. В этих связях таблицы размерности являются родительскими, таблица факта- дочерней. Схема звезда может иметь также консольные таблицы (*outrigger table*), присоединенные к таблице размерности. Консольные таблицы являются родительскими, таблицы размерности - дочерними.

В размерной модели, ERwin обозначает иконкой роль таблицы в схеме звезда:


Таблица факта (fact table)


Таблица размерности (dimensional table),


Консольная таблица (outrigger table).

Прежде чем создать базу данных со схемой типа звезда, необходимо проанализировать бизнес-правила предметной области с целью выяснения центрального вопроса, ответ на который наиболее важен. Все прочие вопросы должны быть объединены вокруг этого основного вопроса и моделирование должно начинаться с этого основного вопроса. Данные, необходимые для ответа на этот вопрос, должны быть помещены в

центральную таблицу модели - таблицу факта (рис.1). В примере, таблица факта содержит суммарные данные о продажах («SALE»), а таблицы размерности содержат данные о заказчике и заказах («CUSTOMER»), продуктах («PRODUCT»), продавцах («SALESPEOPLE») и периодах времени («TIME»).

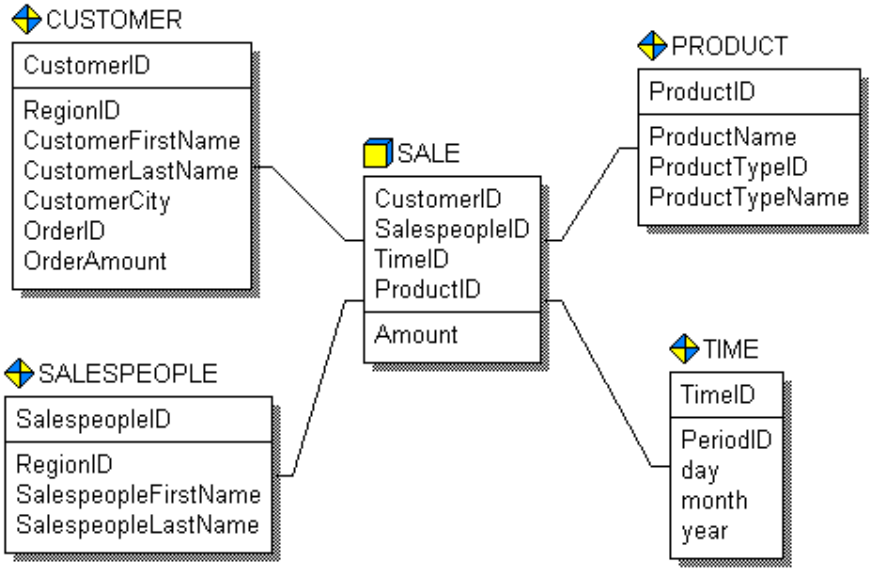

Рис.1. Схема звезда.

Таблица факта является центральной таблицей в схеме звезда, она содержит суммирующие или фактические данные, которые могут помочь ответить на требуемые вопросы. Таблица факта и таблицы размерности связаны идентифицирующими связями, при этом первичные ключи таблицы размерности мигрируют в таблицу факта в качестве внешних ключей. В размерной модели направления связей явно не показываются – они определяются типом таблиц. Первичный ключ таблицы факта целиком состоит из первичных ключей всех таблиц размерности.

Таблицы размерности имеют меньшее количество строк, чем таблицы факта и содержат описательную информацию. Эти таблицы позволяют пользователю быстро переходить от таблицы факта к дополнительной информации.

Консольные таблицы могут быть связаны только таблицами размерности, причем консольная таблица в этой связи родительская, а таблица размерности - дочерняя. Связь может быть идентифицирующей или неидентифицирующей. Консольная таблица не может быть связана с таблицей факта. Она используется для нормализации данных в таблицах размерности. Схема снежинка обычно препятствует эффективности, потому что требует объединения многих таблиц для построения результирующего набора данных, что увеличивает время выполнения запроса (рис. 2). Поэтому при проектировании не следует злоупотреблять созданием множества консольных таблиц.

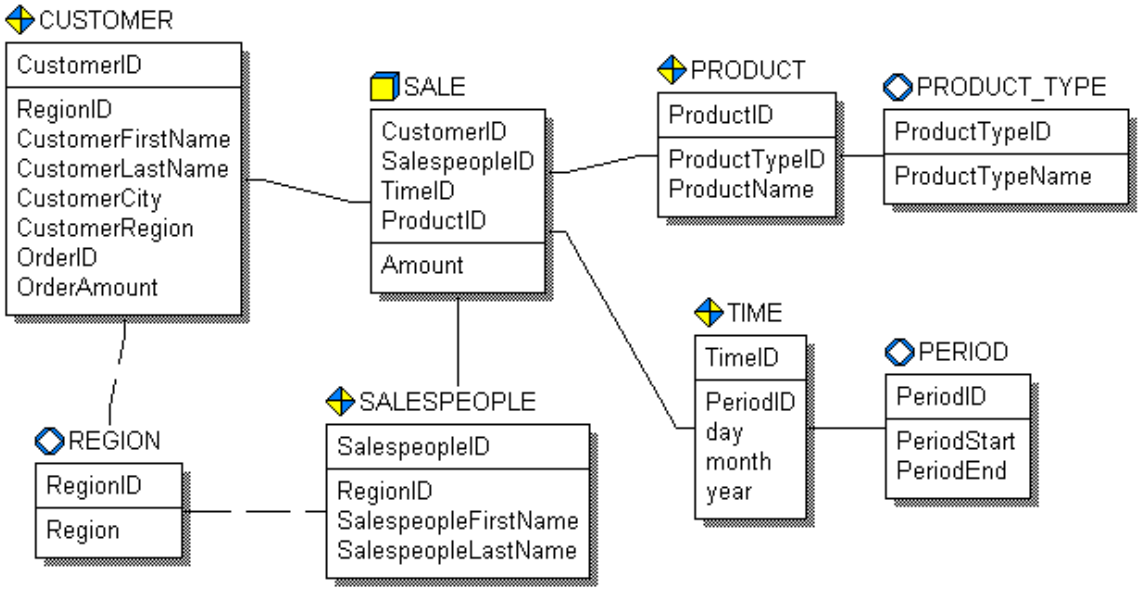

Рис.2. Схема снежинка.

В диалоге описания свойств таблицы Table Editor имеется закладка Dimensional, в которой задаются специфические свойства таблицы в размерной модели (рис. 3):

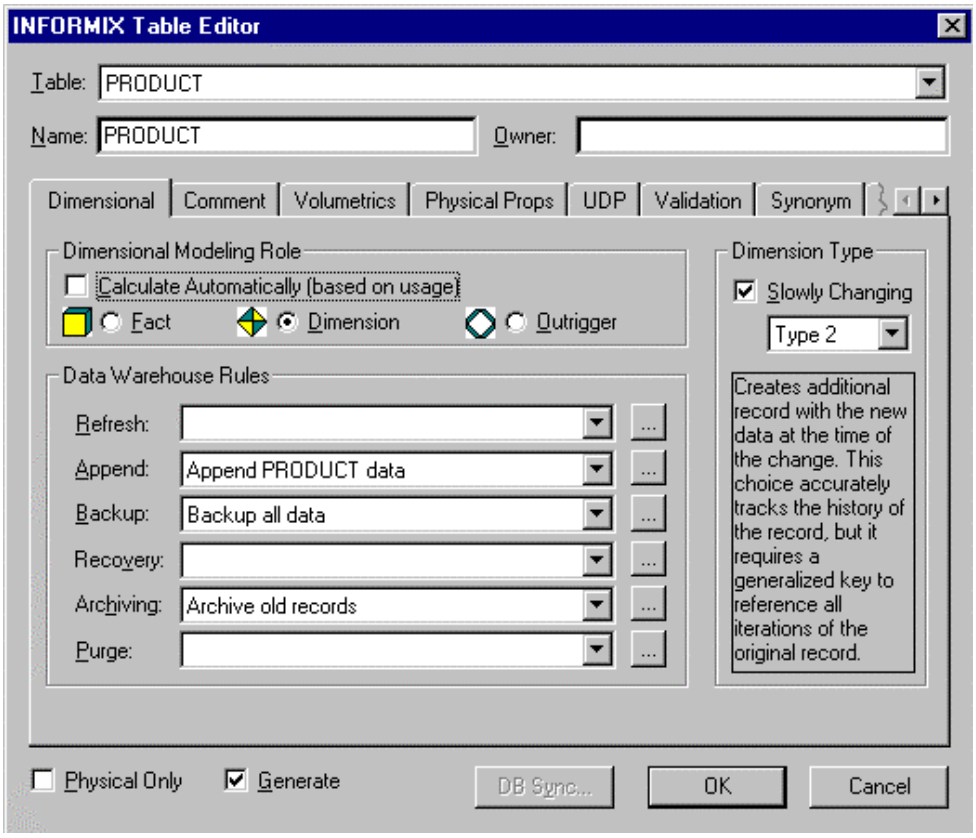

Рис. 3. Закладка Dimensional диалога Table Editor.

Роль таблицы в схеме (Dimensional Modeling Role). По умолчанию Erwin автоматически определяет роль таблицы на основании созданных связей (таблица факта, размерности или консольная). Таблица без связей определяется как таблица размерности,

таблица факта не может быть родительской в связи, таблица размерности может быть родительской по отношению к таблице факта, консольная таблица может быть родительской по отношению к таблице размерности. При ручном назначении роли таблицы ERwin автоматически проверяет корректность размерной модели и выдает диалог с предупреждающим сообщением в случае нарушений синтаксиса.

Тип таблицы размерности (*Dimension Type*). Каждая таблица размерности может содержать неизменяемые, либо редко изменяемые данные (*slowly changing dimensions*). Поскольку хранилище данных имеет ненормализованную структуру, редактирование таблиц размерности может привести к коллизиям. Для того, чтобы избежать противоречий при хранении данных, ERwin позволяет задать тип редко изменяемых данных, который отличается способом редактирования данных:

- Перезаписывание старых данных новыми. При этом старые данные теряются.
- Создание новой записи в таблице размерности с новыми данными и временем изменения. В этом случае сохраняются старые данные и можно проследить историю изменения редактируемых данных, но необходимо генерировать ключ для ссылки на старые данные.
- Запись новых данных в дополнительном поле той же самой записи. В этом случае сохраняется первоначальное и последнее новое значение. Все промежуточные данные теряются.

Правила хранения данных (*Data Warehouse Rules*). Для каждой таблицы можно задать шесть типов правил манипулирования данными: обновление (*Refresh*), дополнение (*Append*), резервное копирование (*Backup*), восстановление (*Recovery*), архивирование (*Archiving*) и очистка (*Purge*). Каждое правило должно быть предварительно описано в диалоге *Data Warehouse Rule Editor* (*Edit→Data Warehouse Rule*). Для каждого правила должно быть задано имя, тип, определение. Связать правила с определенной таблицей можно с помощью диалога *Table Editor*.

При проектировании хранилища данных важно определить источник данных (для каждой колонки), метод, которым исходные данные извлекаются , преобразовываются, и фильтруются прежде, чем они импортируются в хранилище данных. Хранилище данных может объединять информацию из текстовых файлов и многих баз данных, как реляционных так и нереляционных, в единую систему поддержки принятия решений. Чтобы поддерживать регулярные обновления и проверки качества данных, необходимо знать источник для каждой колонки в хранилище данных. Для документирования информации об источниках данных используется редактор *Data Warehouse Source Editor* (рис.4.).

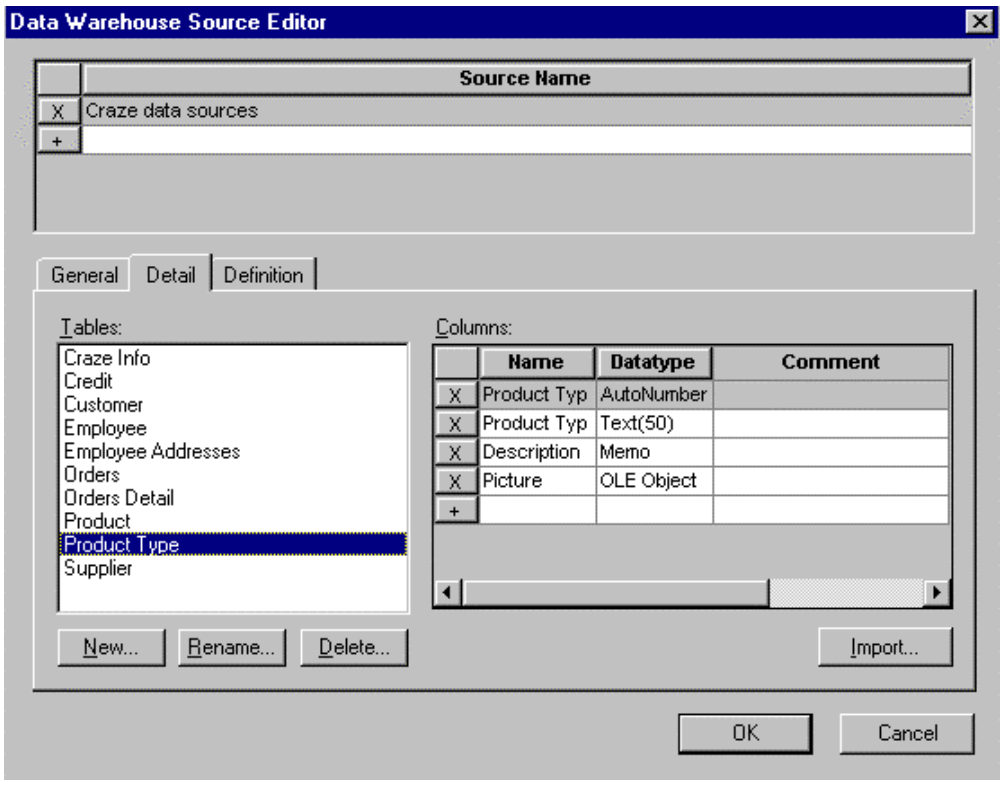

Рис.4. Диалог Data Warehouse Source Editor.

Имена таблиц и колонок источников данных могут быть импортированы как из баз данных (обратное проектирование), так и из других моделей ERwin. Каждому источнику может быть задано имя и определение.

В редакторе *Column Editor* необходимо внести информацию об использовании источников данных для каждой колонки таблиц хранилища данных, а так же дополнительную информацию о способах, режимах и периодичности переноса данных из источника в хранилище данных.

Задание на лабораторную работу

1. Внимательно изучить материалы введения и освоить описанные в нем редакторы ER*win*.
2. Используя как основу модель данных из файла task7-2.er1, построить модель хранилища данных для анализа материалов прессы, полагая, что книги, статьи в газетах и статьи в журналах это одно и то же, но отличающееся друг от друга типом и назначить таблицу, хранящую всю информацию о таком объекте, таблицей фактов. Остальные таблицы модели преобразовать в соответствии с их ролями.
3. Исправить определения элементов модели в соответствиями с новой моделью.
4. С помощью редактора правил хранения данных разработать по два правила для каждого типа операций. Назначить каждое разработанное правило одной из таблиц модели.
5. Сохранить полученную модель под именем **task10-1.er1**.

6. Сгенерировать на основе этой модели базу данных в СУБД Microsoft Access.
7. Разработать новую размерную модель по образцу (см. рис. 2). Для таблицы фактов этой модели разработать по одному правилу хранения данных в каждом типе операций.
8. Разработать два источника данных – один использует БД Борей из примеров к Microsoft Access, а второй – текстовой файл в формате CSV. Предварительно необходимо создать этот текстовой файл с помощью любого текстового редактора.
9. Для каждой консольной таблицы назначить источником данных каждого их атрибута базу данных (связи можно устанавливать произвольно). Для таблицы фактов назначить источником атрибута amount текстовой файл.
10. Сгенерировать на основе этой модели базу данных в СУБД Microsoft Access. Саму модель сохранить как **task10-2.er1**.

Контрольные вопросы

1. В чем отличия хранилищ данных от баз данных? В чем их сходство?
2. Какие методологии применяются для разработки хранилищ данных и каким образом настраиваются эти параметры в ER*win*?
3. Перечислите основные роли таблиц в размерной модели и их особенности?
4. В чем состоит цель указания правил хранения данных и источников данных модели?
5. Какие СУБД поддерживают разработку хранилищ данных и какие СУБД разрешает использовать ER*win* для генерации хранилищ данных?