

Способ организации сущностей и взаимосвязей между ними на уровне хранилища данных

Д. Е. Щедролосьев

Херсонский государственный университет, Украина

Article deals with a concrete model of constructing a data storage level for open informational systems with the three-layer architecture. The author offers an objective model for complex architectural decisions.

1. Общая постановка задачи и её актуальность

При нынешнем лавинообразном росте систем управления предприятиями одним из наиболее актуальных вопросов остается эффективное построение архитектуры корпоративных систем. Под эффективной архитектурой системы следует понимать ряд характеристик. Прежде всего, это удовлетворение требований общего характера, предъявляемым к корпоративным системам, и, в следствие, такая организация структуры, которая способствует динамике строения и развитию системы в целом.

Наиболее широко используемыми и эффективными в данной области стали системы с трехуровневой архитектурой. Требования, предъявляемые к каждому уровню, являются основой, без которой невозможно построение эффективной информационной системы. Несмотря на то, что большее внимание обычно уделяется уровню бизнес логики [2], построение оптимальной модели являя базы данных дает возможность создания по-настоящему открытых систем. Хотя одним из достоинств трехуровневой архитектуры является полная независимость бизнес процессов от структуры и формы хранилища данных, реально информационная система строится на вполне конкретной структуре язы данных[2].

Однако очень часто разработчик СУБД сталкивается с проблемой организации эффективной структуры данных на основе объектной модели. Любая информационная система более-менее крупного предприятия не может обойтись без объектных подходов в процессе проектирования системы. Можно много говорить о преимуществах функциональной (линейной) или объектной модели построения системы, однако, основываясь на опыте проектирования корпоративных информационных систем среднего уровня, можно сказать, что наиболее удачным является именно сочетание нескольких подходов.

2. Нерешенные проблемы и цели работы

Определяя необходимые условия, следует выделить задачи, которые требуют теоретического и практического решения. Так, в основу современной модели данных должны быть положены, на наш взгляд, следующие принципы.

- Объектная организация хранимых данных [3];
- Стандартизация хранения объектов и доступа к ним;

- Открытость в структуре хранения данных;
- Независимость структуры хранения типов и количества хранимых объектов;
- Формализация служебного уровня (безопасность, сервисы и т.д.).

Все эти принципы позволяют по-иному подходить к созданию хранилища на основе реляционной модели. Из всех принципов только последний предполагает использование тяжелой модели хранения, все остальные подразумевают только легкую модель.

3. Предложенный метод и его реализация

Основным фактором оптимальности данной модели является независимость структуры данных от вида хранимых объектов. Попытаемся подробнее рассмотреть структурную модель.

При построении большинства систем проектировщики, конечно, пытаются так или иначе соблюдать данные принципы, однако зачастую сложность заключается в несоответствии инструментальных средств классу решаемых задач.

В данной работе представлен один из способов организации объектной модели на основе реляционной базы данных. Данная структура реализована в MS SQL Server. Особенность данного решения - в попытке использовать наиболее популярные реляционные средства построения баз данных и организовать динамическую модель с высоким уровнем абстракции.

В основе всей модели лежат несколько основных понятий. Базовым понятием является сущность. Сущность - это любой объект, которым оперируют в составе информационной системы. Каждая сущность имеет тип (например, сущность - человек, сущность - сотрудник, сущность - отдел и т.д.). Вопрос отношения того или иного объекта к разряду сущностей и их типизации относится к вопросам объектного проектирования постановки и функциональности систем. В случае построения полной объектной модели, все элементы которой участвуют в процессе функционирования системы, ее можно отнести к сущности и сформировать необходимую типизацию. Все сущности состоят из атрибутов. Под атрибутом понимают любые данные, относящиеся к объекту, т.е. его свойства. Любые сущности могут быть связаны друг с другом. Такой тип связи и отношение будет называться ассоциацией.

Среди всего разнообразия сущностей имеются общие атрибуты, которые можно выделить в отдельную таблицу Entity. Данная таблица является главным и ключевым элементом для построения любого объекта. В Entity могут содержаться основные характеристики сущностей (ID, дата создания, тип сущности и т.д.). Кроме того, данная таблица является уникальной идентификацией всех объектов, что позволяет:

- вести общий журнал ошибок с привязкой к каждой сущности;
- организовать единую систему безопасности, используя вспомогательные элементы;
- вести подробную историю изменения сущностей;
- организовать единый механизм связей между сущностями.

Можно считать, что таблица сущностей содержит абстрактные, родительские объекты самого низкого уровня. Для того, чтобы на ее основании построить черный объект, необходимо организовать таблицу, характеризующую определенный уровень абстракции с соответствующим атрибутом, относящимся только к данной сущности.

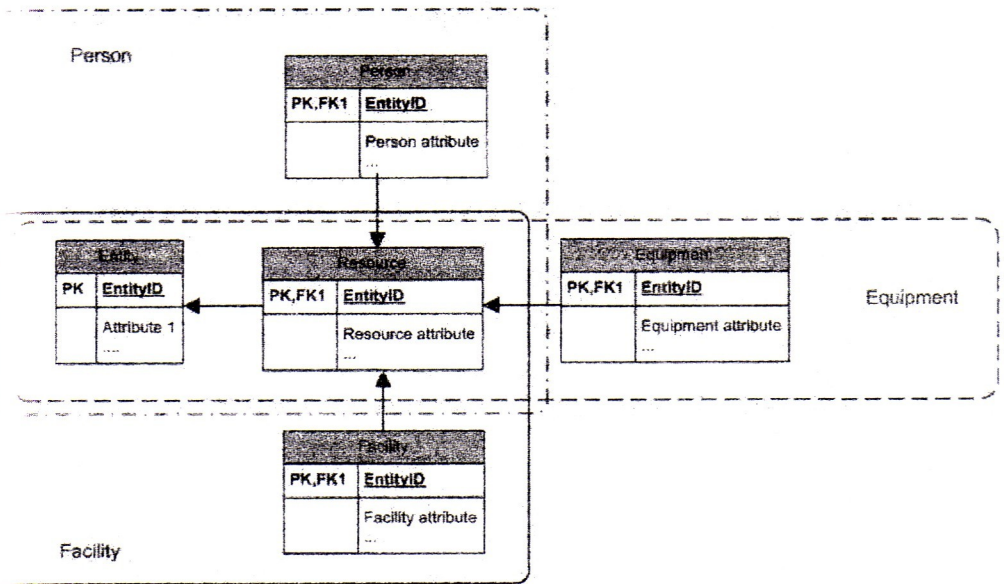


Рис.1. Структура организации сущности.

На схеме приведен пример подобной организации дочерних сущностей. Базовая таблица Entity является хранением родительских объектов. Для хранения объектов Resource используют две таблицы - Entity и Resource, объединение которых включает полный объем атрибутов. Следующей дочерней сущностью может быть Person, Facility и т.д. Полная сущность Person описывается уже тремя таблицами: Entity - Resource - Facility.

Каждый тип сущности описывается вспомогательными таблицами в случае необходимости. Существуют и такие типы сущностей, для которых не нужны дополнительные атрибуты. Например, контейнеры - вспомогательные сущности, которые необходимы для группировки основных сущностей в группы. Таким образом, каждая отдельная сущность представляет собой комбинацию таблиц - абстрактной Entity и специфических, каждая из которых относится конкретному объекту.

В такой структуре актуальным является вопрос организации взаимосвязей между сущностями. Одним из таких методов является древовидный метод. Можно определить в таблице Entity атрибут ParentEntityID, который будет вызывать на родительскую сущность. Поскольку об этом объекте имеется запись в Entity, базовое дерево будет определено однозначно. Однако в большинстве случаев требуется не только древовидное построение взаимосвязей сущностей.

Для организации взаимосвязей между сущностями целесообразно использовать таблицу ассоциаций. Это - вспомогательные элементы, которые имеют определенный тип и две ссылки на таблицу-сущность.

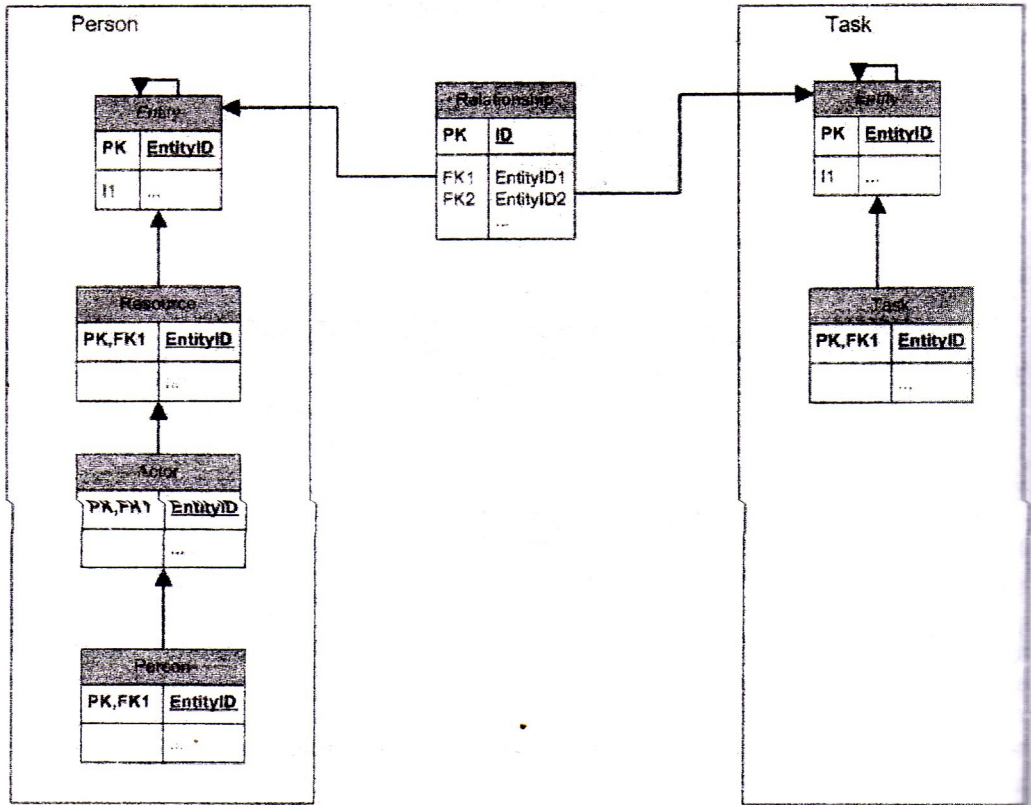


Рис. 2. Взаимосвязи между различными сущностями.

Подобным образом можно описать любое отношение между двумя сущностями.

Все схемы организации данных хорошо ложатся в структуру компонентного подхода при организации уровней бизнес процессов. Хотя целью данной статьи не является раскрытие всей структуры организации бизнес логики, автор считает целесообразным упомянуть о некоторых аспектах в качестве примера эффективного использования объектной структуры данных. Допустим, что для организации бизнес процессов используются соответствующие каждой сущности отдельные объекты. Тогда в структуре данных существует базовый объект Entity, его наследует Resource, Resource наследует Person и т.д. В таком случае, при добавлении нового типа сущности в состав проекта необходимо создать вспомогательную таблицу, расширяющую значения новой сущности, но вся другая структура (проверка безопасности, вызов журнала событий и т.д.) абсолютно не изменяется. При этом, если создается простая сущность, например, контейнер, то не требуется даже расширение таблиц. Достаточно создать новый объект и наследовать базовую сущность.

4. Выводы по результатам

Данный способ организации данных имеет преимущества и недостатки. К преимуществам можно отнести следующие:

- Единая идентификация объектов.
- Возможность единого журнала событий.
- Единая система безопасности сущностей.
- Единая система модификации объектов.
- Общий механизм организации сущностей.
- Возможность простого расширения сущностей.
- Достаточно простые способы создания новых типов сущностей и т.д.

Конечно, наряду с этим существует и ряд недостатков. Наиболее серьезным из них является отсутствие возможностей организации целостности на уровне базы данных. Очевидно, что не всегда данная структура является приемлемой для разработчика базы данных при построении проектов. Способ организации механизмов наследования должен быть прописан достаточно эффективно.

Данная реализация была применена при проектировании VPM - информационной системы управления проектами и работами, которая разработана и эксплуатируется в Херсонском государственном университете. Общая архитектура проекта имеет трехуровневую структуру. В качестве хранилища данных используется MS SQL 2000. Уровень бизнес логики представляет собой .Net компоненты, хотя с точки зрения структуры это не имеет особого значения.

Стоит отметить, что любая модель может быть или не быть эффективной в зависимости от ее реализации и применения.

Выбор правильного подхода зависит как от задания, определяемого рамками проекта, так и от степени подготовленности самого проекта. Модель, эффективная в одном случае, может оказаться абсолютно неприемлемой в другом. В любом проекте необходим оптимальный подход к выбору модели как структуры данных, так и проекта в целом.

ЛИТЕРАТУРА

1. Томас Коннолли, Каролин Бегг, Анна Страчан. Базы данных: проектирование, реализация и сопровождение. – М.: «Вильямс», 2001. – 1112 с.
2. Левашев В. Построение корпоративных систем // Информационная инфраструктура высших учебных заведений: Сб. работ международной научно-практической конференции, Херсон, 2000, - С.12-17
3. Гради Буч Объектно-ориентированное проектирование - К: «Диалектика» и М.: «И.В.К.», 1992.