Лекция 13:

**Метод моделирования "Свод данных"**

**A**

 |

**Аннотация:**В настоящей лекции рассматривается метод моделирования хранилищ данных, который получил название "Свод данных". Объясняются основные понятия метода, приводятся примеры построения логических моделей для "Свода данных".

**Цель лекции**

Изучив материал настоящей лекции, вы будете знать:

* что представляет собой *метод моделирования "Свод данных"* ;
* что такое *сущность-концентратор* в модели "Свод данных";
* что такое *сущность-связь* в модели "Свод данных";
* что такое *сущность-сателлит* в модели "Свод данных";
* что такое *сущность "момент времени"* в модели "Свод данных";
* что такое *сущность-мост* в модели "Свод данных";
* алгоритм моделирования хранилища данных методом "Свод данных";

и научитесь:

* применять алгоритм моделирования хранилища данных методом "Свод данных";
* как формировать *сущности-концентраторы* в модели "Свод данных";
* как формировать *сущности-связи* в модели "Свод данных";
* как формировать *сущности-сателлиты* в модели "Свод данных";
* как заполнять объекты "Свода данных".

**Введение**

**Свод данных** (*Data* Vault) как метод моделирования данных для ХД был предложен в конце 2002 года Dan Linstedt [[57](https://intuit.ru/studies/courses/599/455/literature#literature.57)]. ***Метод моделирования "Свод данных"*** **— это методология проектирования, разработанная для глобальных ХД масштаба предприятия и имеющая в основе набор связанных нормализованных таблиц, ориентированных на поддержку функциональных областей бизнеса с возможностью отражения истории**. Метод удачно сочетает требования нормализации и возможности схемы "*звезда*".

Использование этого метода предполагает наличие у проектировщика ХД базового уровня знаний в области моделирования данных, т.е. понимание таких терминов, как **таблица** (*table*), **взаимосвязь** (*relationship*), **родитель** (*parent*), **потомок** (*child*), **ключ** (primary/foreign key), **измерение** (*dimension*) и **факт** (*fact*).

Исследователи в области обработки данных постоянно ищут структуры данных для приложений искусственного интеллекта (*artificial intelligence* — AI) и *извлечения знаний* (*data mining* — DM). Большинство технологий DM предполагает импорт данных из подающих информационных систем в плоский *файл* (*flat file*) для того, чтобы объединить форму представления данных с функцией *извлечения знаний*. Поскольку объем данных в ХД растет быстро, экспорт информации для приложений DM становится затруднительным. Таким образом, возникает разрыв между формой представления (структурой), функцией (AI) и выполнением (DM).

Такой разрыв между формой, функцией и выполнением снижает эффективность использования методов AI и DM. Поэтому задача разработки структур данных, которые математически позволяют использовать технологии AI непосредственно в базах данных, остается очень актуальной. С точки зрения моделирования структур данных метод *Data* Vault основан на математических принципах, которые позволяют эффективно управлять большими объемами информации. Особенно этот метод эффективен для создания структур данных для динамического управления изменениями во взаимосвязях между данными как единицами представления информации в компьютерных системах. Он позволяет динамически управлять изменением взаимосвязей между данными в системе в процессе эволюции сохраняемых в ней данных.

**Метод моделирования "Свод данных" (Data Vault)**

**Определение метода проектирования "Свод данных" (Data Vault)**

**Свод данных** (Data Vault), **по определению, является ориентированным на детали набором нормализованных связанных таблиц, которые обеспечивают информационную поддержку одной или более предметных областей деятельности организации**. Этот подход является комбинацией методики реляционного проектирования (до *третьей нормальной формы* — *3NF*) и методики многомерного проектирования. *Метод моделирования "Свод данных"* был разработан для создания моделей данных глобальных ХД масштаба предприятия. Он основан на математических принципах, которые поддерживают нормализованные модели данных. По существу модель "Свод данных" соответствует нормализованной до *3NF* схеме "звезда", включая измерения, связи "многие ко многим" и таблицы стандартной структуры. Различие лежит в более детальном представлении взаимосвязей и элементов данных, структурированных и детализованных во временном изменении. Этот метод проектирования был разработан, чтобы объединить гибкость структур обработки данных OLTP-систем с мощностью аналитической обработки данных в OLAP-системах. Он является масштабируемым и легко адаптируемым методом разработки структур данных для решения задач анализа данных в масштабах предприятия.

**Проблемы моделирования данных для хранилищ данных**

Обычно применение известных методик проектирования к разработке модели ХД масштаба предприятия, например, таких как нормализация, сталкивается с рядом трудностей.

В частности, использование *3NF* для структур данных приводит к следующему.

* Временная зависимость в первичном ключе (time-driven primary key) приводит к увеличению сложности поддержки отношения "родитель-потомок" и учету влияния каскадных изменений в таких отношениях.
* Сложно обеспечить высокую производительность при загрузке данных в реальном времени в структуру в *3NF*.
* Во многих случаях усложняется доступ к данным при обработке запросов.
* Возникают проблемы при использовании анализа на основе свертки-развертки данных (*drill*-down analysis).

На [рис. 18.1](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#image.18.1) показана попытка адаптировать структуру данных в *3NF* к использованию в ХД. Одна из проблем этой структуры связана с размещением временной метки (data/*time stamp*) в первичном ключе родительской таблицы, для того чтобы представить изменения детальных данных во времени. Это проблема масштабируемости и гибкости структуры. Если данные добавляются в родительскую таблицу, изменения каскадно распространяются через все *подчиненные таблицы*. Например, когда новая строка вставляется с *родительским ключом* (parent key), у которого изменяется только поле временной метки, все дочерние строки должны быть переназначены на новый родительский ключ. Этот каскадный эффект имеет отрицательное влияние на обработку данных в таких таблицах, причем чем сложнее и больше структура, тем сильнее влияние каскадного эффекта. Для модели данных масштаба предприятия это создает трудности в расширении и сопровождении модели данных, и, как следствие, усложняется процесс проектирования.



**Рис. 18.1.**Временная метка в третьей нормальной форме

Существует проблема и для *взаимосвязанных киосков данных* (*conformed* *data marts*). Такая архитектура глобального ХД представляет собой набор таблиц фактов, которые связаны между собой посредством первичных и внешних ключей или, другими словами, набор взаимосвязанных схем "звезда". При такой реализации ХД возникает ряд проблем, таких как изолированное представление предметно-ориентированных областей, возможное дублирование данных (*data redundancy*), различие представления таблиц фактов по уровню структурированности (детализуемости или *гранулированности) данных*, синхронизация данных во время загрузки в реальном времени, ограниченность использования технологии DM в масштабах предприятия и др. Схема "звезда" является типичной архитектурой, которая проектируется и реализуется по методологии "снизу вверх", и *взаимосвязанные киоски данных* создаются на основе подхода "снизу вверх", а реализуются на основе подхода "сверху вниз".

Одной из наиболее сложных проблем *взаимосвязанных киосков данных* является выбор правильного уровня *гранулированности* данных (*grain*) для таблиц фактов. Это означает, что агрегирование данных во всех таблицах будет согласованным по измерению времени, а структура каждой таблицы фактов не будет изменяться с точки зрения добавления новых измерений. Такой подход к проектированию ограничивает масштабируемость и гибкость модели данных. Другой проблемой могут быть вспомогательные таблицы в измерениях, которые обслуживают ссылки для взаимоотношений между измерениями. *Гранулированность* и стабильность измерений являются важными факторами успешного проектирования ХД.

Например, если *гранулированность* факта "Суммарный доход" таблицы "Суммарный доход" ([рис. 18.2](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#image.18.2)) изменяется, то это должно привести к дублированию таблицы фактов с добавлением дополнительных атрибутов. Предположим, что таблицы фактов связаны между собой только посредством одних и тех же ключей измерений. При добавлении нового измерения к одной из таблиц фактов (например, к таблице фактов "Суммарный доход" добавим измерение "Контракт") факты в таблице "История контрактов" также должны измениться. Таблицы фактов не изменятся, только если они имеют одну и ту же *гранулированность*.



**Рис. 18.2.**Взаимосвязанные киоски данных

Среди практиков-разработчиков ХД сложилось мнение, что архитектура ХД должна проектироваться на основе методологии "сверху вниз", а реализация выполняться на основе методологии "снизу вверх". Такой подход позволяет максимально приблизить архитектуру к пониманию задач предметной области ХД, в то время как реализация может поэтапно включать фрагменты предметной области в общее ХД, не нарушая миссию и видение *системы складирования данных*. Подходы к проектированию и разработке архитектуры ХД должны быть гибкими, чтобы быстро адаптироваться к росту объема данных и расширению или изменению предметных областей в системе.

Одним из подходов к решению задач разработки типовых моделей и архитектур данных является определенная нормализация структур данных. Так же, как и структуры БД OLTP-систем (*1NF*, *2NF*, *3NF*; *4NF*, *5NF*), БД ХД должны иметь определенную степень нормализации структуры данных. Модель "Свод данных" и является одной из таких нормализованных структур данных для ХД. Она включает методы построения структур данных для отношений "многие ко многим", ссылочной целостности, минимизации дублирования данных и установления семантических связей между ключевыми бизнес-функциями предметной области через **концентраторы** (hubs).

**Элементы модели "Свод данных"**

*Модель проектирования* "Свод данных", аналогично методам *многомерного моделирования* или " *сущность-связь* ", содержит ряд структурных компонент, новыми из которых являются *сущности-концентраторы*, или **хабы**, *сущности-связи* и *сущности-сателлиты*. Проектирование этим методом фокусируется на функциональных предметных областях деятельности организации. Каждая такая область характеризуется бизнес-ключом и представляется в концентраторе первичным ключом. *Сущности-связи* обеспечивают интеграцию операций между хабами. *Сущности-сателлиты* обеспечивают контекст первичного ключа хаба. Каждая из этих сущностей сконструирована для обеспечения максимальной гибкости и масштабируемости модели данных ХД масштаба предприятия.

***Сущности-концентраторы (Hub Entities). Сущности-концентраторы***, или просто **хабы** (hubs), **являются таблицей, которая содержит минимальный список бизнес-ключей (натуральных ключей)**. Это ключи, которые используются организацией в каждой ежедневной операции: например, номер счета, табельный номер сотрудника, номер покупателя, номер изделия и номер автомобиля. Если в процессе деятельности такой ключ был потерян, то, как правило, теряются и ссылка на контекст, и сопутствующая информация. Помимо натуральных ключей (на [рис. 18.3](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#image.18.3) – атрибут "Номер покупателя в источнике данных"), концентраторы могут иметь следующие атрибуты:

* **суррогатный ключ** – опциональный атрибут, который является обычно членом числовой последовательности. На [рис. 18.3](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#image.18.3) – атрибут "Номер";
* **временная метка загрузки** (Load Data/*Time Stamp*) – это дата и время, когда ключ впервые появился в БД. На [рис. 18.3](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#image.18.3) — атрибут "Время загрузки из источника";
* **источник данных** (Record Source) – записывается для трассировки данных. На [рис. 18.3](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#image.18.3) — атрибут "Наименование источника данных".



**Рис. 18.3.**Пример концентратора для покупателей

[Рис. 18.3](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#image.18.3) показывает пример сущности "Концентратор для покупателей". В этой сущности атрибут "Номер покупателя в источнике данных" является первичным бизнес- ключом, а атрибут "Номер" является суррогатным ключом, назначенным для покупателей внутри системы. В [табл. 18.1](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#table.18.1) приведен пример контекста для сущности "Концентратор для покупателей".

|  |
| --- |
| Таблица 18.1. Контекст сущности "Концентратор для покупателей" |
| **Номер** | **Номер покупателя в источнике данных** | **Время загрузки из источника** | **Наименование источника данных** |
| 1 | 1234 | 23.01.2009 | Продажи |
| 2 | 1235 | 24.01.2009 | Контракты |
| 3 | 2266 | 26.01.2009 | Финансы |
| 4 | 2344 | 28.01.2009 | Продажи |

*Cущности-концентраторы* не могут быть связаны отношением "один ко многим" (родитель-потомок). Для построения взаимосвязей между концентраторами используются *сущности-связи*.

***Связывающая сущность, или сущность-связь (Link Entitiy)***. *Сущности-связи* **являются физическим представлением взаимосвязи "многие ко многим" в 3NF**. Связь представляет собой взаимоотношение или операцию между двумя или более бизнес-компонентами или бизнес-ключами. *Сущности-связи* содержат следующие атрибуты (см. [рис. 18.4](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#image.18.4)):

* **суррогатный ключ** – опциональный атрибут, который используется при связывании более двух концентраторов (на [рис. 18.4](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#image.18.4) не показан);
* **ключи концентраторов** (Hub Key) – ключи концентраторов, которые мигрируют в *сущность-связь* для формирования *составного ключа*, связывающего эти концентраторы;
* **временная метка загрузки** (Load Data/*Time Stamp*) – дата и время записи связи в БД;
* **источник данных** (Record Source) – используется для трассировки данных.



**Рис. 18.4.**Пример сущности-связи "Покупатель Счет"

Этот компонент модели предназначен для разрешения проблемы отношения "многие ко многим" для ХД. Вместе с *сущностями-концентраторами* *связывающие сущности* описывают поток данных предметной области ХД. [Табл. 18.2](https://intuit.ru/studies/courses/599/455/lecture/10179?page=1#table.18.2) иллюстрирует содержание соответствующих сущностям таблиц БД.

|  |
| --- |
| Таблица 18.2. Таблицы базы данных, представляющие концентраторы и сущность-связь |
| **Концентратор для покупателей** |
| **Номер** | **Номер покупателя в источнике данных** | **Время загрузки из источника** | **Наименование источника данных** |
| 1 | 1234 | 23.01.2009 | Продажи |
| 2 | 1235 | 24.01.2009 | Контракты |
| **Связывающая сущность** |
| **Идентификатор покупателя** | **Идентификатор счета** | **Время загрузки из источника** | **Наименование источника данных** |
| 1 | 100 | 25.01.2009 | Продажи |
| 2 | 200 | 26.01.2009 | Контракты |
| **Концентратор для счетов** |
| **Номер** | **Номер счета в источнике данных** | **Время загрузки из источника** | **Наименование источника данных** |
| 100 | 12/124 | 25.01.2009 | Продажи |
| 200 | 12/135 | 26.01.2009 | Контракты |

Следующий компонент модели отвечает за контекст, а именно отвечает на вопросы, когда, почему, что, где и кто создает операции и бизнес-ключи (методика 5W). Например, знание номера автомобиля не является поводом для его покупки покупателем. Покупателю нужно знать цвет, марку и т.д. Для хранения такой информации служат *сущности-сателлиты*.

***Сущности-сателлиты (Satellite Entities)***. *Сущности-сателлиты* **содержат описательную информацию о ключах концентраторов, а именно когда, почему, что, где и кто создает операции и бизнес-ключи**. Например, в отличие от номера автомобиля, его цвет, марка и т.д. могут изменяться во времени, и, следовательно, структура данных должна отражать эти изменения на каждом уровне структурирования информации (*гранулированности*).

*Сущности-сателлиты* обычно содержат следующие атрибуты:

* **первичный ключ** — первичный ключ концентратора или связи, который мигрировал в сателлит;
* **временная метка загрузки** (Load Data/*Time Stamp*) – это дата и время записи описательной информации в БД;
* **источник данных** (Record Source) – записывается для трассировки данных;
* *сущности-сателлиту* может быть назначен **суррогатный первичный ключ**.



**Рис. 18.5.**Сущность-сателлит для адреса покупателя

На [рис. 18.5](https://intuit.ru/studies/courses/599/455/lecture/10179?page=2#image.18.5) приведена *сущность-сателлит*, содержащая информацию об изменении адреса клиента во времени. Атрибут "Время загрузки" является частью составного первичного ключа. Поскольку данные упорядочены по времени, основное назначение *сущности-сателлита* — *обеспечить описание ключа концентратора*. [Табл. 12.3](https://intuit.ru/studies/courses/599/455/lecture/10179?page=2#table.18.3) иллюстрирует содержание соответствующих сущностям таблиц БД.

|  |
| --- |
| Таблица 18.3. Таблицы базы данных, представляющие концентратор и сущность-сателлит |
| **Концентратор для покупателей** |
| **Номер** | **Номер покупателя в источнике данных** | **Время загрузки из источника** | **Наименование источника данных** |
| 1 | 1234 | 23.01.2009 | Продажи |
| 2 | 1235 | 24.01.2009 | Контракты |
| **Сателлит для адреса покупателей** |
| **Номер покупателя** | **Время загрузки** | **Наименование источника** | **Адрес** |
| 1 | 23.01.2009 | Паспорт | ул. Первая, д.1, кв. 1 |
| 1 | 15.03.2009 | Паспорт | Институтский пр., д. 6, кв. 3 |
| 2 | 24.01.2009 | Паспорт | ул. Вторая, д. 3, кв. 2 |
| 2 | 14.02.2009 | Паспорт | ул. Коммунальная, д. 2, кв. 4. |

***Сущность "Момент времени" (Point-In-Time)*** **является производной от сателлита**. Она строится для обеспечения поиска информации для заданных моментов времени. Пример *сущности "Момент времени"* приведен на [рис. 18.6](https://intuit.ru/studies/courses/599/455/lecture/10179?page=2#image.18.6) ниже. [Табл. 18.4](https://intuit.ru/studies/courses/599/455/lecture/10179?page=2#table.18.4) иллюстрирует содержание соответствующих сущностям таблиц БД (структура таблиц упрощена в целях наглядности).



**Рис. 18.6.**Пример сущности "Момент времени"

|  |
| --- |
| Таблица 18.4. Таблицы базы данных, представляющие *сущность "Момент времени"*, "Изменение данных о покупателе" – концентратора для покупателей и его сателлитов |
| **Изменение данных о покупателе** |
| **Номер покупателя** | **Время загрузки изменений** | **Время загрузки адреса** | **Время загрузки имени** |
| 1 | 23.01.2009 | 23.01.2009 | 23.01.2009 |
| 1 | 15.03.2009 | 15.03.2009 | 15.03.2009 |
| 1 | 31.04.2009 | 31.04.2009 | 15.03.2009 |
| **Концентратор для покупателей** |
| **Номер** | **Номер покупателя в источнике данных** | **Время загрузки из источника** | **Наименование источника данных** |
| 1 | 1234 | 23.01.2009 | Продажи |
| **Сателлит для адреса покупателей** |
| **Номер покупателя** | **Время загрузки** | **Наименование источника** | **Адрес** |
| 1 | 23.01.2009 | Паспорт | Ул. Первая, д.1, кв. 1 |
| 1 | 15.03.2009 | Паспорт | Институтский пр., д. 6, кв. 3 |
| **Сателлит для имени покупателя** |
| **Номер покупателя** | **Время загрузки** | **Наименование источника** | **Фамилия** |
| 1 | 23.01.2009 | Паспорт | Прохоров |

*Сущность "Момент времени"* является производной от сателлита. Она строится для обеспечения поиска информации для заданных моментов времени.

***Сущность-мост (Bridge)*** содержит временные метки последней загрузки. Эта сущность подобна *сущности "Момент времени"*, но охватывает всю предметную область или схему данных.

Бизнес-пользователи часто хотят видеть данные, сгруппированные различным образом. В простейшем случае одно подразделение (к примеру, отдел маркетинга) имеет свою иерархию покупателей, а другое подразделение (к примеру, отдел продаж) имеет другую иерархию тех же покупателей. Можно включить обе иерархии в измерение "Покупатель". Однако несколько иерархий, встроенных прямо в измерение, сделают его малопригодным для использования.

Бизнес-требование более гибкой реализации дополнительных иерархий возникает тогда, когда нескольким подразделениям необходимо группировать одни и те же данные по различным схемам классификации, причем в нескольких разных вариантах. В таком случае необходимо поработать с пользователями и определить наиболее распространенную группировку данных. Эта группировка станет стандартной иерархией, используемой по умолчанию, и будет встроена прямо в основную таблицу измерения. Также можно поступить еще с несколькими наиболее широко используемыми иерархиями для простоты работы пользователей.

Для поддержки дополнительных иерархий в ХД создается отдельная таблица, с помощью которой пользователь может сгруппировать данные по любой из имеющихся иерархий. Это и есть *сущность-мост*, или *таблица-мост* (bridge table). На [рис. 18.7](https://intuit.ru/studies/courses/599/455/lecture/10179?page=2#image.18.7) показан пример такой промежуточной таблицы "Географическое положение покупателя" (CustomerRegionHierarchy) для группировки по географическому положению.



**Рис. 18.7.**Пример сущности-моста

В запросе значение поля "Имя иерархии" ( **HierarchyName** ) задает необходимую группировку данных. Например, предикат в предложении WHERE HierarchyName = 'Отдел маркетинга' задает группировку данных для отдела маркетинга.

Каждая иерархия в промежуточной таблице должна быть полной, т.е. начинаться с того уровня базового измерения, к которому присоединена промежуточная таблица, и заканчиваться самым верхним уровнем. Например, таблица "Географическое положение покупателя" (CustomerRegionHierarchy) соединена с уровнем "Область" (State).

Для упрощения анализа и создания отчетов промежуточная таблица должна содержать описание и стандартной иерархии. Стандартная иерархия становится используемой по умолчанию во всех предварительно настроенных отчетах, но пользователю предоставляется возможность переключиться на другую иерархию. Отдельная таблица "Иерархия" (Hierarchy) с одной строкой на каждую иерархию упрощает поддержку системы, но визуально усложняет дизайн. При необходимости можно провести денормализацию и объединить таблицы "Иерархия" (Hierarchy) и "Географическое положение покупателя" (CustomerRegionHierarchy).

***Сущность-связь для группировки пользователей (User Grouping Link)*** содержит информацию, которая предоставляет пользователю определенную точку зрения на данные, но не влияет на содержание данных в ХД. [Таблица 18.5](https://intuit.ru/studies/courses/599/455/lecture/10179?page=2#table.18.5) показывает, как используется *сущность-связь для группировки пользователей*.

|  |
| --- |
| Таблица 18.5. Использование сущности-связи для группировки пользователей (ключевые атрибуты выделены жирным шрифтом) |
| **Таблица измерения "Группы покупателей"** |
| **ID** | **Метка группировки** | **Дата загрузки** | **Описание источника** |
| 1 | Крупные покупатели | 10.04.2009 | Excel |
| 2 | Мелкие покупатели | 12.04.2009 | Excel |
| **Сущности-связи для группировки пользователей** |
| **Номер группы** | **Номер покупателя** | **Дата загрузки** | **Описание источника** |
| 1 | 100 | 14.04.2009 | Excel |
| 2 | 101 | 14.02.2009 | Excel |
|  |
| **ID** | **Номер покупателя** | **Дата загрузки** | **Описание источника** |
| 100 | 100ADB12 | 14.04.2009 | Finance |
| 101 | *ADF*-1 | 14.04.2009 | Finance |

Таким образом, *сущность-связь для группировки пользователей* позволяет устанавливать, как они будут разворачивать данные таблиц фактов.

Мы рассмотрели основные и дополнительные элементы модели "Свод данных" и можем перейти к описанию общего алгоритма построения модели ХД описанным выше методом.

**Алгоритм построения модели "Свод данных" (Data Vault)**

При создании модели "Свод данных" необходимо сначала создать сущности и описать их атрибуты, а затем установить связи между ними. Сущности должны создаваться в следующем порядке.

1. *Сущности-концентраторы*, которые должны содержать бизнес-ключи предметной области.
2. *Сущности-связи* для поддержки взаимосвязей между бизнес-ключами, т. е. информация о деятельности организации в контексте бизнес-ключей.
3. *Сущности-сателлиты* для описания полной картины деятельности организации с точки зрения бизнес-процедур.
4. *Сущности "Момент времени"* для обеспечения поиска моментов времени изменения описательной информации.

При создании связей в структуре модели "Свод данных" следует соблюдать правила *поддержки ссылочной целостности* (*referential integrity*).

1. Ключи концентраторов не могут мигрировать в другие концентраторы, т.е. не поддерживается отношение "родитель-потомок" для концентраторов.
2. Концентраторы взаимодействуют между собой через *сущности-связи*.
3. *Сущность-связь* может быть использована для связи более двух концентраторов.
4. *Сущности-связи* могут быть связаны друг с другом.
5. *Сущности-связи* должны связывать, по крайней мере, два концентратора.
6. Суррогатные ключи могут использоваться для концентраторов и связей.
7. Ключи концентраторов всегда мигрируют.
8. Бизнес-ключи концентраторов никогда не изменяются, так же, как и их первичные ключи.
9. Сателлиты могут связываться с концентраторами и *сущностями-связями*.
10. Могут быть использованы стандартные таблицы временных меток (standalone table), такие как календарь, время, код и описание.
11. Сателлиты всегда содержат либо временную метку загрузки, либо числовой указатель на временную метку загрузки (на стандартные таблицы временных меток)
12. *Сущности-связи* могут иметь суррогатные ключи.
13. Если концентратор имеет более двух сателлитов, то может быть создана *сущность "Момент времени"*.
14. В сателлитах не должно быть строк-дубликатов.
15. Данные в сателлитах разделяются по типу информации или по скорости изменения.

Изменения в данных собираются в сателлитах. Если размер сателлитов растет очень быстро, то можно создать два новых сателлита, чтобы ограничить такой процесс роста. Данные в новых сателлитах могут разделяться по типу информации или по скорости изменения.

Концентраторы хранят бизнес-ключи основных направлений деятельности организации (предметных областей). Обычно бизнес-ключи меняются очень редко. Первичный ключ концентратора используется в *сущностях-связях*, чтобы представить операции основных направлений деятельности.

Теперь рассмотрим применение вышеизложенного алгоритма на учебном примере, т.е. построим модель "Свод данных" для схемы учебного примера.

**Пример проектирования модели "Свод данных"**

**Учебная база данных**

Рассмотрим БД Northwind, которая разработана Microsoft в учебных целях. Ее модель данных приведена на [рис. 18.8](https://intuit.ru/studies/courses/599/455/lecture/10179?page=3#image.18.8).

Как видно из рисунка, для этой модели характерно использование нестандартных типов данных: bit, ntext, image, money. Использование нестандартных типов данных может привести к проблемам преобразования данных, если для создания ХД будет использована другая СУБД, чем в OLTP-системе. В нашем случае "Свод данных" будет построен на основе той же СУБД, что и OLTP, а именно для MS SQL Server 2008 компании MicroSoft.



[увеличить изображение](https://intuit.ru/EDI/20_07_20_2/1595197216-9970/tutorial/632/objects/18/files/12_08.jpg)
**Рис. 18.8.**Схема учебной БД в третьей нормальной форме

**Процесс моделирования "Свода данных"**

Рассмотрим процесс преобразования нормализованной модели примера ([рис. 18.7](https://intuit.ru/studies/courses/599/455/lecture/10179?page=2#image.18.7)) в модель "Свод данных". Процесс преобразования, как было рассмотрено в предыдущих разделах, включает следующие этапы.

1. Идентификация бизнес-ключей и создание набора суррогатных ключей, формирование концентраторов модели.
2. Идентификация взаимосвязей между таблицами, которые должны поддерживаться, формирование *сущностей-связей* модели.
3. Идентификация описательной информации, формирование сателлитов модели.
4. Нормализация (перегруппировка) сателлитов либо по скорости изменения, либо по типу информации. Добавление *сущностей-мостов* и *сущностей "Момент времени"*.

Отметим сразу же, что для модели учебного примера добавление *сущностей-мостов* и *сущностей "Момент времени"* не требуется.

В случае работы более чем с одной моделью данных (интеграция нескольких источников данных) преобразование следует начитать с модели главной с точки зрения направлений деятельности организации системы, а затем поэтапно рассматривать модели других подсистем с целью получения унифицированной точки зрения на представление данных в модели системы в целом.

Теперь перейдем к реализации пунктов 1-3 процесса формирования модели "Свод данных" для учебного примера.

**Формирование сущностей-концентраторов**

Будем следовать рассмотренному нами процессу построения модели "Свод данных". Сначала идентифицируем бизнес-ключи и поместим их в *сущности-концентраторы* стандартной структуры. Так как концентраторы являются по определению списком бизнес-ключей, важно разместить их вместе с суррогатными ключами, если такие существуют в модели.

Проведя исследование модели на [рис. 18.8](https://intuit.ru/studies/courses/599/455/lecture/10179?page=3#image.18.8) (уникальные индексы, запросы к данным и т.д.), мы сможем построить следующие группы бизнес-ключей/суррогатных ключей и определим сущности, претендующие на роль концентраторов в модели "Свод данных".

Таблица "Категории" ( **Categories** ) имеет бизнес-ключ "Имя категории" ( **CategoryName** ) и суррогатный ключ "Идентификатор категории" ( **CategoryID** ). Они будут составными элементами *сущности-концентратора* "Концентратор\_Категория" ( **HUB\_Category** ).

Таблица "Товары" ( **Products** ) имеет бизнес-ключ "Наименование товара" ( **ProductName** ) и суррогатный ключ "Идентификатор товара" ( **ProductID** ). Они будут составными элементами *сущности-концентратора* "Концентратор\_Товар" ( **HUB\_Product** ).

Таблица "Поставщики" ( **Suppliers** ) имеет бизнес-ключ "Наименование поставщика" ( **SupplierName** ) и суррогатный ключ "Идентификатор поставщика" ( **SupplierID** ). Они будут составными элементами *сущности-концентратора* "Концентратор\_Поставщик" ( **HUB\_Supplier** ).

Таблица "Позиции заказа" ( **Order Details** ) не имеет бизнес-ключа, она не представляет бизнес-процесс и, следовательно, не может иметь свой концентратор.

Таблица "Заказы" ( **Orders** ) имеет суррогатный ключ, который может быть, а может и не быть связан с бизнес-ключом. Это зависит от бизнес-требований. Эта таблица является транзакционной по своей природе и является кандидатом более на *сущность-связь*, чем на концентратор.

Таблица "Грузоперевозчики" ( **Shippers** ) имеет бизнес-ключ "Наименование компании" ( **CompanyName** ) и суррогатный ключ "Идентификатор грузоперевозчика" ( **ShipperID** ). Они будут составными элементами *сущности-концентратора* "Концентратор\_Перевозчик" ( **HUB\_Shippers** ). Заметим, что если бизнес-требования требуют интеграции компаний-грузоперевозчиков, то поле "Наименование компании" ( **CompanyName** ) может быть использовано как бизнес-ключ. Однако если бизнес-требования требуют, чтобы грузоперевозчики поддерживались в системе отдельно друг от друга, то указанное поле не является достаточно описательным для бизнес-процесса и должно быть заменено полем "Наименование грузоперевозчика" ( **ShipperName** ).

Таблица "Покупатели" ( **Customers** ) имеет бизнес-ключ "Наименование компании" ( **CompanyName** ) и суррогатный ключ "Идентификатор покупателя" ( **CustomerID** ). Они будут составными элементами *сущности-концентратора* "Концентратор\_Покупатели" ( **HUB\_Customers** ). Обратим внимание на то, что если нужна интеграция покупателей и грузоперевозчиков, то концентратор может быть назван "Концентратор\_Компания" ( **HUB\_Company** ), чтобы интегрировать грузоперевозчиков и покупателей.

Таблица "Покупатель\_Покупатель" ( **CustomerCustomerDemo** ) не имеет бизнес-ключа и, следовательно, не может быть преобразована в концентратор. Эта таблица является кандидатом на *сущность-связь*.

Таблица "Демография покупателей" ( **CustomerDemographics** ), на первый взгляд, имеет бизнес-ключ **CustomerDesc** и суррогатный ключ **CustomerTypeID**, и для нее может быть создан концентратор "Концентратор\_Демография\_покупателей" ( **HUB\_CustomerDemographics** ). Однако отметим, что эта таблица может рассматриваться и как *сущность-сателлит* для покупателей.

Таблица "Служащие" ( **Employees** ) имеет бизнес-ключ "Имя служащего" ( **EmployeeName** ) и суррогатный ключ "Идентификатор служащего" ( **EmployeeID** ). Они будут составными элементами *сущности-концентратора* "Концентратор\_Служащий" ( **HUB\_Employee** ).

Таблица "Служащий Территория" ( **EmployeeTerritories** ) не имеет бизнес-ключа и, следовательно, не может быть преобразована в концентратор. Эта таблица является кандидатом на *сущность-связь*.

Таблица "Территория" ( **Territories** ) имеет бизнес-ключ "Описание территории" ( **TerritoryDescription** ) и суррогатный ключ "Идентификатор территории" ( **TerritoryID** ). Они будут составными элементами *сущности-концентратора* "Концентратор\_Территория" ( **HUB\_Territories** ).

Таблица "Регион" ( **Region** ) имеет бизнес-ключ "Описание региона" ( **RegionDescription** ) и суррогатный ключ "Идентификатор региона" ( **RegionID** ). Они будут составными элементами *сущности-концентратора* "Концентратор\_Регион" ( **HUB\_Region** ).

Сейчас мы можем сформировать список *сущностей-концентраторов* модели, которые должны быть построены ([рис. 18.9](https://intuit.ru/studies/courses/599/455/lecture/10179?page=3#image.18.9)).

* "Концентратор\_Категория" - Hub\_Categories
* "Концентратор\_Товар" - Hub\_Producst
* "Концентратор\_Поставщик" - Hub\_Suppliers
* "Концентратор\_Перевозчик" - Hub\_Shippers
* "Концентратов\_Компания" - Hub\_Customers
* "Концентратор\_Демография\_покупателей" - Hub\_CustomerDemographics
* "Концентратор\_Служащий" - Hub\_Employees
* "Концентратор\_Территория" - Hub\_Territories
* "Концентратор\_Регион" - Hub\_Region

|  |  |
| --- | --- |
| "Концентратор\_Категория" | Hub\_Categories |
| "Концентратор\_Товар" | Hub\_Producst |
| "Концентратор\_Поставщик" | Hub\_Suppliers |
| "Концентратор\_Перевозчик" | Hub\_Shippers |
| "Концентратов\_Компания" | Hub\_Customers |
| "Концентратор\_Демография\_покупателей" | Hub\_CustomerDemographics |
| "Концентратор\_Служащий" | Hub\_Employees |
| "Концентратор\_Территория" | Hub\_Territories |
| "Концентратор\_Регион" | Hub\_Region |



**Рис. 18.9.**Идентификация сущностей-концентраторов модели "Свод данных" для учебного примера

Для обозначения бизнес-ключей будем использовать иные наименования, чем на схеме [рис. 18.8](https://intuit.ru/studies/courses/599/455/lecture/10179?page=3#image.18.8).

Рассмотрим список концентраторов на [рис. 18.9](https://intuit.ru/studies/courses/599/455/lecture/10179?page=3#image.18.9). Обратим внимание на тот факт, что суррогатные ключи на схеме [рис. 18.8](https://intuit.ru/studies/courses/599/455/lecture/10179?page=3#image.18.8) однозначно определяют бизнес-ключи соответствующих таблиц. Следовательно, мы можем заменить в концентраторах бизнес-ключи на соответствующие им суррогатные ключи. Это целесообразно сделать потому, что размеры суррогатных ключей, как правило, значительно меньше, чем размеры соответствующих бизнес-ключей. Так, тип суррогатного ключа "Идентификатор категории" ( **CategoryID** ) есть int, что составляет 2 байта, в противоположность 16 байтам бизнес-ключа "Имя категории" (CategoryName) (в концентраторе поле **CTO\_NAME** ). Это обычная практика при проектировании ХД.

На [рис. 18.10](https://intuit.ru/studies/courses/599/455/lecture/10179?page=3#image.18.10) приведен окончательный список *сущностей-концентраторов* для модели учебного примера.

Таблица концентратора для *сущности-концентратора* создается командой CREATE TABLE, как показано для концентратора "Концентратор\_Категории" ( **Hub\_Categories** ) ниже. Аналогично в БД создаются остальные концентраторы модели.

CREATE TABLE Hub\_Categories (

 CategoryID int NOT NULL,

 CTO\_LOAD\_DTS datatime NOT NULL,

 CTO\_REC\_SRC nvarchar(20) NOT NULL,

PRIMARY KEY (CTO\_SEQ\_ID)

);

CREATE UNIQUE INDEX Hub\_Categories\_idx

ON Hub\_Categories (CategoryID);



**Рис. 18.10.**Список сущностей-концентраторов модели "Свод данных" для учебного примера

Теперь мы можем перейти к формированию *сущностей-связей* для модели "Свод данных" учебного примера.

**Формирование сущностей-связей**

На втором этапе преобразования исходной модели в "Свод данных" необходимо идентифицировать *сущности-связи*, или взаимосвязи между бизнес-ключами. *Сущности-связи* представляют бизнес-процессы организации, их предназначение - связать бизнес-ключи между собой. Это наиболее важные элементы ХД в модели "Свод данных". Без них трудно связывать данные между собой. Транзакционные таблицы и таблицы для разрешения связей "многие ко многим" являются хорошими кандидатами в *сущности-связи*.

Для модели учебного примера можно выделить следующие *сущности-связи*.

Таблица "Позиции заказа" ( **OrderDetails** ) находится в отношении "многие ко многим", и поэтому для нее будет построена *сущность-связь* "Связь\_Позиции\_заказа" ( **LNK\_OrderDetails** ).

Таблица "Заказы" ( **Orders** ) является родительской таблицей для таблицы "Позиции заказа" **OrderDetails**, поэтому для нее будет построена *сущность-связь* "Связь\_Заказы" ( **LNK\_Orders** ).

Таблица "Покупатель Покупатель" ( **CustomerCustomerDemo** ) находится в отношении "многие ко многим", и поэтому для нее будет построена *сущность-связь* "Связь\_Покупатель\_Покупатель" ( **LNK\_CustomerCustomerDemo** ).

Таблица "Служащий Территория" ( **EmployeeTerritories** ) находится в отношении "многие ко многим", и поэтому для нее будет построена *сущность-связь* "Связь\_Служащий\_Территория" ( **LNK\_EmployeeTerritories** ).

Таблица "Территория" ( **Territories** ) находится в отношении "многие ко многим", и поэтому для нее будет построена *сущность-связь* "Связь\_Территория" ( **LNK\_Territories** ).

Продолжим исследование модели учебного примера. Нетрудно заметить, что некоторые таблицы, являющиеся кандидатами в *сущности-концентраторы*, находятся в отношении "родитель-потомок" по отношению к внешнему ключу. Для таблицы "Товары" ( **Products** ) это ключи "Идентификатор категории" ( **CategoryID** ) и "Идентификатор поставщика" ( **SupplierID** ). Поэтому целесообразно построить таблицу **LNK\_Products** *сущности-связи*, включающую "Идентификатор товара" ( **ProductID** ), "Идентификатор поставщика" ( **SupplierID** ) и "Идентификатор категории" ( **CategoryID** ). Заметим, что для этой таблицы не требуется суррогатный ключ, поскольку "Идентификатор товара" ( **ProductID** ) является достаточным для представления поставщиков и категорий.

Таблица "Служащие" ( **Employees** ) имеет рекурсивное отношение. Для его представления введем таблицу *сущности-связи* "Связь\_Служащие" ( **LNK\_ Employees** ).

Других взаимосвязей, которые можно вынести в *сущности-связи* в модели учебного примера, больше нет. Теперь мы можем объединить *сущности-концентраторы* и *сущности-связи* в схеме модели, как показано на [рис. 18.11](https://intuit.ru/studies/courses/599/455/lecture/10179?page=4#image.18.11).



[увеличить изображение](https://intuit.ru/EDI/20_07_20_2/1595197216-9970/tutorial/632/objects/18/files/12_11.jpg)
**Рис. 18.11.**Идентификация сущностей-концентраторов и сущностей-связей модели "Свод данных" для учебного примера

Таблица БД для *сущности-связи* создается командой CREATE TABLE, как показано для связи "Связь\_Товары" ( **LNK\_Products** ) ниже. Аналогично в БД создаются остальные *сущности-связи* модели.

CREATE TABLE LNK\_Products (

ProductID int NOT NULL,

CategoryID int NOT NULL,

SupplierID int NOT NULL,

CTO\_LOAD\_DTS datatime NOT NULL,

CTO\_REC\_SRC nvarchar(20) NOT NULL,

PRIMARY KEY (ProductID),

FOREING KEY(SupplierID) REFERENCES HUB\_Suppliers,

FOREING KEY(CategeoryID) REFERENCES HUB\_Categories

);

Построив *сущности-концентраторы* и *сущности-связи*, мы можем перейти к формированию *сущностей-сателлитов* для модели "Свод данных" учебного примера.

**Формирование сущностей-сателлитов**

Оставшиеся поля таблиц схемы [рис. 18.8](https://intuit.ru/studies/courses/599/455/lecture/10179?page=3#image.18.8) являются экземплярами сущностей, которые изменяются во времени и, следовательно, будут размещены в *сущностях-сателлитах* х. В качестве *сущностей-сателлитов* в рассматриваемой модели могут выступать следующие таблицы:

* таблица "Категории" (Categories) - *сущность-сателлит* **CAT\_CATEGORIES** ;
* таблица "Товары" (Products) - *сущность-сателлит* **CAT\_PRODUCTS** ;
* таблица "Поставщики" (*Suppliers*) - *сущность-сателлит* **CAT\_SUPPLIERS** ;
* таблица "Заказы" (Orders) - *сущность-сателлит* **CAT\_ORDERS** ;
* таблица "Покупатели" (Customers) - *сущность-сателлит* **CAT\_CUSTOMERS** ;
* таблица "Грузоперевозчики" (Shippers) - *сущность-сателлит* **CAT\_SHIPPERS** ;
* таблица "Служащие" (Employees) - *сущность-сателлит* **SAT\_EMPLOYEES** ;
* таблица "Территории" (Territories) - *сущность-сателлит* **SAT\_TERRITORIES** ;
* таблица "Регион" (Region) - *сущность-сателлит* **SAT\_REGION** ;
* таблица "Демография покупателей" (CustomerDemographics) - *сущность-сателлит* **SAT\_CUSTOMERDEMOGRAPHICS**.

|  |  |
| --- | --- |
| таблица "Категории" (Categories) | *сущность-сателлит* **CAT\_CATEGORIES** ; |
| таблица "Товары" (Products) | *сущность-сателлит* **CAT\_PRODUCTS** ; |
| таблица "Поставщики" (*Suppliers*) | *сущность-сателлит* **CAT\_SUPPLIERS** ; |
| таблица "Заказы" (Orders) | *сущность-сателлит* **CAT\_ORDERS** ; |
| таблица "Покупатели" (Customers) | *сущность-сателлит* **CAT\_CUSTOMERS** ; |
| таблица "Грузоперевозчики" (Shippers) | *сущность-сателлит* **CAT\_SHIPPERS** ; |
| таблица "Служащие" (Employees) | *сущность-сателлит* **SAT\_EMPLOYEES** ; |
| таблица "Территории" (Territories) | *сущность-сателлит* **SAT\_TERRITORIES** ; |
| таблица "Регион" (Region) | *сущность-сателлит* **SAT\_REGION** ; |
| таблица "Демография покупателей" (CustomerDemographics) | *сущность-сателлит* **SAT\_CUSTOMERDEMOGRAPHICS**. |

Заметим, что *сущности-сателлиты* не содержат атрибутов, зависящих от внешнего ключа. Первичный ключ *сущности-сателлита* является *составным ключом*: первичный ключ соответствующей *сущности-концентратора* и время загрузки данных.

Теперь мы можем объединить *сущности-сателлиты* с *сущностями-концентраторами* и *сущностями-связями* в схеме модели, как показано на [рис. 18.12](https://intuit.ru/studies/courses/599/455/lecture/10179?page=4#image.18.12).



**Рис. 18.12.**Идентификация сущностей-концентраторов, сущностей-связей и сущностей-сателлитов модели "Свод данных" для учебного примера



[увеличить изображение](https://intuit.ru/EDI/20_07_20_2/1595197216-9970/tutorial/632/objects/18/files/12_12_1.jpg)
**Рис. 18.12.**Идентификация сущностей-концентраторов, сущностей-связей и сущностей-сателлитов модели "Свод данных" для учебного примера

Таблица БД для *сущности-сателлита* создается командой CREATE TABLE, как показано для сателлита "Сателлит\_Товары" ( **CAT\_Products** ) ниже. Аналогично в БД создаются остальные *сущности-сателлиты* модели.

CREATE TABLE SAT\_Products (

ProductID int NOT NULL,

PRD\_LOAD\_DTS DateTime NOT NULL,

QuantityPerUnit nvarchar(20),

UnitPrice money,

UnitsInStock smallint,

UnitsOnOrder smallint,

ReOrderLevel smallint,

Discontinued bit,

PRD\_REC\_SRC nvarchar(20) NOT NULL,

PRIMARY KEY (ProductID, PRD\_LOAD\_DTS)

FOREING KEY (ProductID) REFERENCES HUB\_Products

);

Таким образом, мы завершили построение модели "Свод данных" для схемы данных учебного примера.

Теперь мы можем перейти к обсуждению вопросов о том, как заполнять объекты полученной физической модели.

**Заполнение данными объектов "Свода данных"**

На практике для заполнения объектов "Свода данных" целесообразно использовать виртуальные таблицы или представления, по одному на каждый *объект* модели.

В *сущности-концентраторы* данные только вставляются: при создании ХД заносятся бизнес-ключи. *Сущности-связи* заполняются аналогичным образом. В *сущности-сателлиты* данные вставляются, когда происходят изменения данных в системах источниках.

При загрузке *сущности-концентратора* выбираются только те бизнес-ключи (вместе с их суррогатными ключами), которых еще нет в концентраторе. Для реализации этого процесса можно использовать *представление*, приведенное ниже, на примере "Концентратора\_Категории".

CREATE VIEW V\_INS\_HUB\_CATEGORIES AS

SELECT DISTINCT

A.CATEGORYID,

GETDATE() LOAD\_DATE,

'NORTHWIND' RECORD\_SOURCE

FROM NORTHWIND..[CATEGORIES] A with (NOLOCK)

WHERE NOT EXISTS

(SELECT \* FROM HUB\_CATEGORIES WITH (NOLOCK))

При загрузке *сущности-связи* выбираются только те составные ключи (вместе с их суррогатными ключами), которых еще нет в связи. Для реализации этого процесса можно использовать *представление*, приведенное ниже, на примере "Связь\_Заказы".

CREATE VIEW V\_INS\_LNK\_ORDERS AS

SELECT DISTINCT

A.ORDERID,

A.CUSTOMERID,

A.EMPLOYEEID,

A.SHIPVIA,

GETDATE() LOAD\_DATE,

'NORTHWIND' RECORD\_SOURCE

FROM NORTHWIND..[ORDERS] A with (NOLOCK)

WHERE NOT EXISTS

(SELECT \* FROM LNK\_ORDERS WITH (NOLOCK))

При загрузке *сущности-сателлита* выбирается набор записей, соответствующий или соединенный по бизнес-ключу (или по составному ключу), колонки которых имели изменения в таблицах источниках данных для сателлита, при этом выбираются только самые поздние изменения. Для реализации этого процесса можно использовать *представление*, приведенное ниже, на примере "Сателлит\_Служащие".

CREATE VIEW V\_UPD\_SAT\_EMPLOYEES AS SELECT

A.EMPLOYEEID,A.LASTNAME, A.FIRSTNAME, A.TITLE, A.TITLEOFCOURTESY,

A.BIRTHDATE,

A.HIREDATE, A.ADDRESS, A.CITY, A.REGION, A.POSTALCODE, A.COUNTRY,

A.HOMEPHONE, A.EXTENSION, A.PHOTO, A.NOTES, A.REPORTSTO,

A.PHOTOPATH, GETDATE() LOAD\_DATE, 'northwind' RECORD\_SOURCE

FROM northwind..[employees] A with (NOLOCK),

SAT\_EMPLOYEES B with (NOLOCK)

WHERE A.EMPLOYEEID = B.EMPLOYEEID

AND (isnull(A.LASTNAME,'x') != isnull(B.LASTNAME,'x')

OR isnull(A.FIRSTNAME,'x') != isnull(B.FIRSTNAME,'x')

OR isnull(A.TITLE,'x') != isnull(B.TITLE,'x')

OR isnull(A.TITLEOFCOURTESY,'x') != isnull(B.TITLEOFCOURTESY,'x')

OR isnull(A.BIRTHDATE,convert(datetime,'01/01/1960')) !=

 isnull(B.BIRTHDATE,convert(datetime,'01/01/1960'))

OR isnull(A.HIREDATE,convert(datetime,'01/01/1960')) !=

 isnull(B.HIREDATE,convert(datetime,'01/01/1960'))

OR isnull(A.ADDRESS,'x') != isnull(B.ADDRESS,'x')

OR isnull(A.CITY,'x') != isnull(B.CITY,'x')

OR isnull(A.REGION,'x') != isnull(B.REGION,'x')

OR isnull(A.POSTALCODE,'x') != isnull(B.POSTALCODE,'x')

OR isnull(A.COUNTRY,'x') != isnull(B.COUNTRY,'x')

OR isnull(A.HOMEPHONE,'x') != isnull(B.HOMEPHONE,'x')

OR isnull(A.EXTENSION,'x') != isnull(B.EXTENSION,'x')

OR isnull(CONVERT(varbinary(2000),A.PHOTO),0) != isnull(CONVERT(varbinary(2000),B.PHOTO),0)

OR isnull(CONVERT(varchar(2000),A.NOTES),'x') != isnull(CONVERT(varchar(2000),B.NOTES),'x')

OR isnull(A.REPORTSTO,0) != isnull(B.REPORTSTO,0)

OR isnull(A.PHOTOPATH,'x') != isnull(B.PHOTOPATH,'x')

)

Представления работают хорошо, когда *БД*-источник и "Свод данных" являются сущностями одной реляционной *БД*. Если это не так, то возможны два решения: 1) применение промежуточной области (*stage*) для источника данных так, чтобы при этом представления могли быть использованы; 2) применение ETL-инструментов для преобразования, сравнения и загрузки данных.

**Резюме**

"Свод данных" есть предметно-ориентированный, поддерживающий историю и уникальные связи в данных набор нормализованных таблиц для обеспечения информационной поддержки одного или нескольких направлений хозяйственной деятельности организации.

"Свод данных" имеет три основных "строительных блока".

* *Сущность-концентратор (Hub)* – таблица, содержащая список уникальных бизнес-ключей. Используется для идентификации направлений хозяйственной деятельности организации.
* *Сущность-связь (Link)* – таблица связей, содержащая уникальные взаимосвязи между ключами. Используется для фиксации взаимосвязей между *сущностями-концентраторами* и *сущностями-связями*.
* *Сущность-сателлит (Satellite)* – таблица, содержащая описательные и исторические данные. Используется для хранения описательной информации для *сущностей-концентраторов* и *сущностей-связей*.

В общих чертах *алгоритм* построения "Свода данных" состоит в проектировании сущностей в следующем порядке:

* *сущности-концентраторы*, которые должны содержать бизнес-ключи предметной области;
* *сущности-связи* для поддержки взаимосвязей между бизнес-ключами, т. е. схема деятельности организации в контексте бизнес-ключей;
* *сущности-сателлиты* для описания полной картины деятельности организации с точки зрения бизнес-процедур.

Одним из главных преимуществ метода "Свод данных" является динамическое *представление* взаимосвязей *предметной области* ХД. Взаимосвязи определяются через бизнес-ключи концентраторов и фиксируются в *сущностях-связях*. Они существуют во времени, и их история сохраняется в сателлитах. Это позволяет отображать динамику развития взаимосвязи.

Метод "Свод данных" целесообразно использовать в следующих случаях:

1. для создания динамических ХД (Dynamic Data Warehousing), когда возникает необходимость учитывать динамику изменения как обработки данных, так и структур данных;
2. для создания Data mining/Exploration Warehousing, когда пользователям нужно менять структуру данных без потери информации;
3. при встраивании процедур DM в ХД.

Таким образом, мы рассмотрели еще один метод моделирования ХД.