**Лекция 7:**

**Моделирование темпоральных (временных) данных в хранилищах данных**

**Цель лекции**

Изучив материал настоящей лекции, вы будете знать:

* что такое *темпоральные (временные) данные* ;
* что представляет собой логическая *модель темпоральных данных* ;
* что такое *временная метка*, *моментная временная метка*, *интервальная временная метка* ;
* что такое *таблицы моментальных снимков*, *таблицы событий*, *таблицы состояний* ;
* какова семантика темпоральных запросов;
* что такое *классы временной зависимости* ;

и научитесь:

* создавать *таблицы моментальных снимков*, *таблицы событий*, *таблицы состояний* ;
* разрабатывать темпоральные запросы;
* разрабатывать логические *темпоральные модели данных* ;
* преобразовывать *темпоральные модели данных*.

**Литература**: [[3](https://intuit.ru/studies/courses/599/455/literature#literature.3)], [[37](https://intuit.ru/studies/courses/599/455/literature#literature.37)].

### Темпоральные данные и базы данных

Целью настоящего раздела является введение в **темпоральные базы данных** (*temporal* *database*). В русскоязычной литературе вместо термина "темпоральные *базы данных*" иногда применяется термин "временные *базы данных*". Поскольку в этой области информационных технологий отсутствует устоявшаяся русскоязычная терминология, далее будем пользоваться первым термином. В этой лекции мы будем говорить о реляционной модели данных и ее темпоральных расширениях. Следует отметить, что данные в компьютерных системах так или иначе связаны с различными событиями, интервалами времени. Однако лишь незначительная часть проектировщиков и разработчиков понимает, что это отдельная и вполне самостоятельная область исследований. Зачастую темпоральные системы создаются лишь на основе общих методов проектирования и разработки приложений баз данных, без использования накопленных знаний в области создания систем управления темпоральными *БД*. Как правило, такой подход не только повышает *стоимость* разработки, но и может самым негативным образом сказаться на эффективности работы приложений и наличии в них ошибок. Поэтому остановимся подробней на представлении *темпоральных данных* в *БД*.

Совсем не сложно указать бизнес-приложения, которые используют данные, зависящие от времени. Например, финансовые приложения, *информационные системы управления* полетами самолетов, *информационные системы управления* отелями, научные приложения (*мониторинг* окружающей среды) оперируют данными, которые меняются во времени. *Список* может быть продолжен. *СУБД* в таких системах управляют данными, которые ссылаются на время, и следовательно, время связывается с сущностями, которые хранятся в базах данных.

Что же такое ***темпоральные данные***? В очень широком смысле – **это произвольные данные, которые явно или неявно связаны с определенными датами или промежутками времени**. Под такое *определение* попадают почти любые данные и *информация*. Например, даже если нет явной зависимости от времени для какого-нибудь факта или события, то все равно для него имеется неявная зависимость от времени, так как когда-то нам (или системе) стало известно, что такой факт существует. Кроме того, есть *вероятность*, что в будущем факт будет пересмотрен или на условия его использования будут наложены определенные ограничения; поэтому нельзя уже будет рассматривать его как некоторую абсолютную истину, верную во всех ситуациях и в любой момент времени.

**Темпоральные базы данных – это базы данных, хранящие темпоральные данные**. Такие *БД* и содержащиеся в них данные могут рассматриваться как темпоральные только в том случае, если известно правило интерпретации *временных меток* и интервалов для конкретной системы управления базами данных (*СУБД*). Чтобы определить, является ли рассматриваемая *СУБД* темпоральной в полном смысле этого слова, необходимо понять, можно ли отдельно выделить и специальным образом интерпретировать данные атрибута "время". В категорию темпоральных *СУБД* не будут попадать обычные реляционные *СУБД*, в которых поддерживаются связанные со временем типы данных, но интерпретацией и связью данных (или событий) между собой с учетом времени приходится заниматься разработчику. В темпоральной *СУБД* учитываются специфическая природа времени и изменчивость данных с течением времени.

История систем реляционных *БД* насчитывает более четырех десятилетий. Примерно на 10 лет позже появились идеи реализации систем управления темпоральными *БД*, до сих пор не воплощенные полностью ни в одной реализации крупных коммерческих *СУБД*. Принято считать, что впервые идеи управления *темпоральными данными* появились в работе Якова Бен-Зви (Jacob Ben-Zvi) в 1982 г. [[70](https://intuit.ru/studies/courses/599/455/literature#literature.70)]. В 80-е годы прошлого века начали появляться работы по темпоральной логике и использованию данных, зависимых от времени, их представлению внутри системы и визуализации для пользователя. С тех пор предлагались различные модели, создавались прототипы систем темпоральных *БД* ([[69](https://intuit.ru/studies/courses/599/455/literature#literature.69)]).

В темпоральных *БД* хранятся данные, изменяющиеся с течением времени. С другой стороны, многие понимают, что в языке запросов *SQL* отсутствует адекватная и эффективная *поддержка* работы с темпоральными *БД*. Традиционная реляционная *БД* хранит информацию лишь о текущем состоянии, и *СУБД* не предоставляет возможности работать с данными, привязанными к определенным датам или интервалам времени (то есть *темпоральными данными* ). Поэтому почти во всех *БД* *поддержка* работы с такими данными обеспечивается усилиями программистов и администраторов, которые используют для решения задач громоздкие конструкции языка *SQL*.

В реляционной *БД* может сохраняться *информация* о событиях и интервалах времени, соответствующих различным представлениям и связям. Если обработкой подобных данных занимается сам *пользователь*, то используемый тип времени можно назвать *временем, определяемым пользователем*. Его отличительным признаком служит отсутствие интерпретации со стороны *СУБД*, так как обработка данных, связанных со временем, полностью возлагается на пользователя. Фактически, все современные *СУБД* обеспечивают поддержку подобной разновидности времени, например, с помощью введения *специальных типов данных* *DATA* или *TIMESTAMP*.

В общем случае можно выделить два вида данных для представления времени: **время фиксации определенного события или факта** и **время выполнения какого-либо действия или операции**.

Если рассматривать данные, представленные в *БД*, как некоторое отражение текущего состояния действительности для моделируемого мира, то каждая *запись* может восприниматься как некоторый факт, который является истинным в определенный момент или *интервал* времени. При переходе к темпоральной *БД* для каждого факта можно указать тот промежуток времени, когда этот факт являлся истинным в моделируемом мире, представленном в базе данных. Поэтому подобное *представление* времени, когда с данными связывается промежуток времени их актуальности (с точки зрения моделируемого мира), называется ***временем фиксации факта*** (*valid* *time*). Его значения можно сравнить с показаниями часов моделируемого мира. Поскольку довольно часто в *БД* отражается именно реальный мир, могут быть заданы соотношения между значениями времени реального мира и представленной в *БД* моделью. Отметим, что значениями данного типа времени могут быть моменты времени как в прошлом, так и в будущем. Кроме того, эти значения могут изменяться, то есть истинность факта в определенные моменты времени может приниматься или отклоняться.

Например, любое предложение естественного языка, которое имеет значением истину или *ложь* и представляет собой, по сути, факт, связано со временем, когда этот факт становится истинным (в прошлом, настоящем или будущем). Хотя все факты имеют время их фиксации, совершенно не обязательно отражать такое время в структурах данных *БД*.

Другим типом времени, который рассматривается в темпоральных *БД*, является **время операции**, или **транзакционное время**. В любой *СУБД* каждой записи *БД* можно сопоставить тот промежуток времени, когда данная *запись* была представлена в *БД*, т.е. промежуток времени между моментами добавления записи и ее удаления из *БД*. При этом отметим, что операция обновления, которая действительно вносит изменения в *запись*, понимается как составная операция удаления старой записи и добавления новой. Очевидно, что значения времени *операции* не могут относиться к будущему. В подавляющем числе *СУБД* время *операции* используется журналом восстановления системы для работы с блокировками. В некоторых системах администраторы даже могут применять специальные расширения языка *SQL*, позволяющие получить *доступ* ко времени *операции* и истории изменений записей в *БД*.

Таким образом, момент времени, когда факт актуализируется в *БД*, является ***временем выполнения операции***. Время *операции* может быть связано с любой сущностью *БД*, а не только с фактами. Хотя все сущности могут быть связаны с временем *операции*, проектировщик может решить не рассматривать временной аспект для некоторых сущностей *БД*. Например, время размещения набора экземпляров сущности в *БД* за одну операцию вставки (*интервал* времени) может и не учитываться в модели данных. Время *операции*, которое отражает время изменения состояния объектов *БД* или приложений, записывается как *значение* атрибута сущности *БД*.

Чтобы ответить на вопрос, как соотносятся между собою *время фиксации факта* и *время выполнения операции*, рассмотрим следующий пример. Пусть имеется *таблица* *БД*, в которой хранится *информация* о текущей зарплате сотрудника организации ([табл. 7.1](https://intuit.ru/studies/courses/599/455/lecture/10165?page=1#table.7.1)).

|  |  |  |
| --- | --- | --- |
| Таблица 7.1. Таблица БД без темпоральных расширений | | |
| **Сотрудник** | **Отдел** | **Зарплата** |
| Прохоров А.И. | ОВИР | 15000 |
| Соловьева М.Е. | ОВИР | 15000 |
| Амосова Е.С. | ОВИР | 6000 |

При наличии поддержки времени фиксации мы могли бы в любой момент сказать, какая у сотрудника была зарплата за произвольный период времени ([табл. 7.2](https://intuit.ru/studies/courses/599/455/lecture/10165?page=1#table.7.2)). Таким образом, данные о зарплате могут быть представлены как последовательность изменяющихся во времени значений.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 7.2. Таблица БД с поддержкой времени фиксации факта | | | |
| **Сотрудник** | **Отдел** | **Зарплата** | **Время фиксации факта** |
| Прохоров А.И. | ОВИР | 15000 | с 1 июня 2008 |
| Соловьева М.Е. | ОВИР | 15000 | с 1 июня 2008 |
| Амосова Е.С. | ОВИР | 12000 | С 1 января 2008 по 30 июня 2008 |
| Амосова Е.С. | ОВИР | 12000 | с 1 июня 2008 |

При наличии поддержки времени *операции* мы могли бы сказать, в какой момент в таблицу были внесены изменения ([табл. 7.3](https://intuit.ru/studies/courses/599/455/lecture/10165?page=1#table.7.3)).

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 7.3. Таблица БД с поддержкой времени операции | | | |
| **Сотрудник** | **Отдел** | **Зарплата** | **Время операции** |
| Прохоров А.И. | ОВИР | 15000 | с 1 июня 2008 |
| Соловьева М.Е. | ОВИР | 15000 | с 1 июня 2008 |
| Амосова Е.С. | ОВИР | 12000 | с 1 января 2008 по 30 июня 2008 |
| Амосова Е.С. | ОВИР | 12000 | с 1 июня 2008 |

Теперь предположим, что для таблицы поддерживается как *время фиксации факта*, так и время *операции* ([табл. 7.4](https://intuit.ru/studies/courses/599/455/lecture/10165?page=1#table.7.4)). Тогда в случае если неправильно введенные данные были впоследствии исправлены, можно будет точно сказать, когда это было сделано. Кроме того, *информация* о подобных изменениях необходима, так как некорректные данные могли бы быть уже использованы в каких-нибудь отчетах. Поэтому в данном случае требуется *поддержка* времени *операции*. При обновлении значений в системе *интервал* времени также обновляется, поэтому можно просмотреть *список* изменений в *БД*.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 7.4. Таблица БД с поддержкой обоих типов представления времени | | | |
| **Сотрудник** | **Зарплата** | **Время фиксации факта** | **Время операции** |
| Прохоров А.И. | 15000 | с 1 июня 2008 | с 1 июня 2008 |
| Соловьева М.Е. | 15000 | с 1 июня 2008 | с 1 июня 2008 |
| Амосова Е.С. | 12000 | С 1 января 2008 по 30 июня 2008 | с 1 января 2008 по 30 июня 2008 |
| Амосова Е.С. | 12000 | с 1 июня 2008 | с 1 июня 2008 |

Говоря о типах времени, введем понятие о *гранулярности* времени. ***Гранулярность*** **времени показывает, насколько близкие моменты на оси времени все еще будут отличимыми друг от друга**. Например, возможно, что для данных о заработной плате сотрудника достаточно использования *времени фиксации факта* разбиения по дням, а для времени *операции* может быть использовано *разбиение* по секундам, если в *СУБД* возможна более частая *фиксация транзакций*.

В общем случае с каждым типом времени может быть еще связан некоторый календарь, который определяет диапазоны значений, *гранулярность*, соответствия и преобразования между моментами времени для различных осей времени.

Выше при обсуждении *времени фиксации факта* говорилось, что существует некоторый *интервал*, в котором определенный факт являлся истинным. Это так называемое **интервальное представление**. Однако можно рассматривать отдельный момент времени и все факты, которые были истинны в этот конкретный момент ( **точечное представление** ). Здесь говорится о представлении времени с точки зрения пользователя, то есть тех условных моделях, в рамках которых могут формулироваться запросы и возвращаться их результаты. При использовании любого из этих представлений истинность фактов не меняется, но в случае точечного представления мы получаем срез всех фактов на какой-то конкретный момент времени, а для интервального представления нас интересует определенный факт и периоды его истинности. Если говорить об обычной реляционной модели, то она опирается на точечное *представление* для актуального состояния данных.

И со *временем фиксации факта*, и со временем *операции* связывается *домен* "Время", который может быть как дискретным, так и непрерывным. Для представления времени в *БД* обычно применяется *домен* с конечными и дискретными значениями. Предполагается, что значения времени в домене упорядочены.

### Введение в моделирование темпоральных данных

Далее в настоящих лекциях для представления объектов модели *предметной области* на *ER-диаграммах* будет использована *нотация* методологии информационного проектирования (IE). В этой нотации сущность на ER-диаграмме представляется прямоугольником с именем в верхней части. В прямоугольнике перечисляются атрибуты сущности; при этом атрибуты, составляющие *уникальный идентификатор сущности*, подчеркиваются. Для атрибутов может быть задан *домен*.

*Отношение* (*связь*) сущностей на ER-диаграмме изображается линией, соединяющей эти сущности. *Степень связи* изображается с помощью символа "птичья лапка", указывающего на то, что в связи участвует много (N) экземпляров сущности, и одинарной горизонтальной чертой, указывающей на то, что в связи участвует один *экземпляр сущности*. Необязательный *класс принадлежности сущности* к связи изображается с помощью кружочка на линии отношения рядом с сущностью, обязательный *класс* принадлежности – с помощью вертикальной черты на линии отношения рядом с сущностью. *Отношение* читается вдоль линии либо слева направо, либо справа налево.

На [рис. 7.1](https://intuit.ru/studies/courses/599/455/lecture/10165?page=2#image.7.1) приведен пример ER-диаграммы в нотации IE.

[](https://intuit.ru/EDI/20_07_20_2/1595197216-9970/tutorial/632/objects/7/files/06_01.jpg)

**Рис. 7.1.**ER-диаграмма в нотации методологии информационного проектирования (IE)

*Моделирование* *темпоральных данных* использует совокупность методов моделирования для создания моделей временных и исторических данных. ***Модель темпоральных данных*** **— это модель данных, которая состоит из элементов данных и внутренних структур, отражающих изменения элементов модели во времени и фиксирующих те моменты времени, когда эти изменения происходят**. Будем называть модель **битемпоральной**, если в ней присутствуют время *операции* и *время фиксации факта*.

*Моделирование* *темпоральных данных* является сложной задачей для информационных систем, основанных на реляционных *БД*.

Во-первых, такие данные не очень удобно представлять в двумерной реляционной модели. Обработка временных данных требует критериев поиска по диапазону, что приводит к использованию соединений таблиц, которые строятся на использовании неравенств. *Производительность* выполнения таких соединений является проблемой в реляционных *БД*. Во-вторых, для временных данных часто требуется соединять таблицы на основе перекрытия диапазонов дат. В *SQL* нет *операции*, позволяющей непосредственно задать такое соединение, и приходится применять процедурный подход к реализации таких запросов.

Приведем пример *темпоральных данных* ([табл. 7.5](https://intuit.ru/studies/courses/599/455/lecture/10165?page=2#table.7.5)). Пусть требуется отслеживать динамику цен на некоторые товары. Цены на товары в любой момент времени могут измениться, поэтому необходимо хранить *действительный* *диапазон* дат по каждой цене и по каждому товару.

Просматривая таблицу, легко сможем найти, когда какая цена на данный *товар* действовала, а также увидеть, что в период с 26.12.2000 по 01.01.2001 цена на данный *товар* не была установлена. В рассматриваемом примере *цена товара* обладает характеристикой, которая называется действительностью по дате, т.е. цена на *товар* действительна только в определенный период времени. При проектировании *темпоральных моделей данных* необходимо учитывать подобные характеристики данных.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 7.5. Изменение цен на товар | | | |
| **№** | **Дата начала действия новой цены** | **Дата окончания действия цены** | **Цена, руб.** |
| 1 | 01.01.2000 | 15.06.2000 | 2000 |
| 2 | 16.06.2000 | 25.12.2000 | 1998 |
| 3 | 01.01.2001 | 31.12.2001 | 1800 |
| 4 | 01.01.2002 | 25.03.2002 | 1802 |
| 5 | 26.03.2002 | 30.09.2002 | 1700 |
| 6 | 01.10.2002 | 12.11.2002 | 1750 |
| 7 | 13.11.2002 | 31.12.2002 | 1850 |
| 8 | 01.01.2003 |  | 1900 |

Таким образом, *темпоральная модель данных* отличается от традиционных моделей данных наличием одного существенного элемента — **времени**. Для представления времени в *темпоральной модели данных* используются так называемые *временные метки* (*time stamp*). ***Временные метки*** **— это атрибуты, которые связаны с фиксацией показаний времени, например даты события, такого как продажа конкретного товара**. Исследование *временных меток* *предметной области* является одной из самых главных задач проектировщика при моделировании временных данных, потому что *временные метки* могут иметь несколько значений, т.е. неоднозначную интерпретацию в рамках *предметной области*.

При моделировании временных данных обычно применяют следующие интерпретации *временных меток*.

* Время фиксации события или факта (Valid time), т.е. так называемая *действительная дата* — это *временная метка*, которая представляет время события или *состояния предметной области*, как уже было указано в предыдущем разделе. Например, это дата подписи на документе, показания времени, снятые с контролирующих приборов, дата отгрузки проданного товара со склада и т.д.
* Время операции (Transaction time) — это *временная метка*, представляющая время, когда была выполнена операция хозяйственной деятельности организации, как уже было указано в предыдущем разделе.
* Время сбора данных (Capture time) — это *временная метка*, представляющая время, когда данные были извлечены или собраны из источника данных (возможно, некоторой внешней БД).
* Время актуализации данных (Apply time) — это *временная метка*, связанная со временем загрузки данных в ХД.
* ***время, определяемое пользователем*** (User-defined time) – это *временная метка*, представляющая момент или моменты времени, которые пользователь намерен хранить в атрибуте сущности, но непосредственно не связанные с фиксацией временной зависимости в модели данных. Например, атрибуты "День рождения", "День государственного праздника" и т.п.

*Темпоральные данные* в ХД будут представлены колонками таблиц с типом "дата/время". Поскольку *интерпретация* *временных меток* в рамках *предметной области* может быть различной, каждая из таких интерпретаций порождает свой собственный *домен* в реляционной *БД*. *Анализ* данных часто использует *операции* поиска и выборки данных на соединениях таблиц. Если в таких операциях соединения будут использоваться *временные метки*, то их домены должны совпадать. Поэтому проектировщик ХД должен очень внимательно определить *временные метки* ХД, т.е. изучить их интерпретацию и соответствующим образом определить для каждой метки свой *домен*. Проектировщик фиксирует интерпретацию *временных меток* ХД в **метаданных**, о которых речь пойдет дальше.

Существует два основных типа *временных меток* — *моментные временные метки*, или **временная метка события** (Instant *timestamp*), и **временные метки диапазона**, или *интервальные временные метки* (*Interval* *timestamp*).

***Моментная временная метка*** **— это временная метка с одним значением, которое представляет момент времени**. Такие метки используются для представления в ХД объектов, изменяющихся во времени, таких как *операции* или события хозяйственной деятельности. Также такие *временные метки* применяются для фиксации изменения значений данных, извлекаемых из оперативных систем и затем сохраняемых в ХД.

***Интервальные временные метки*** **представляют собой период времени или диапазон и содержат обычно два значения: начала и окончания периода или длины интервала**. Такие *временные метки* используются обычно для моделирования состояния объекта и его поведения во времени. В случае если *длина* интервала является фиксированной, например, день или месяц, то для представления *интервальной временной метки* можно задействовать одно *значение*.

На [рис. 7.2](https://intuit.ru/studies/courses/599/455/lecture/10165?page=2#image.7.2) показаны примеры определения сущностей с *моментной временной меткой* (Сущность\_1) и *интервальными временными метками* (Сущность\_2).



**Рис. 7.2.**Определение сущностей с моментными и интервальными временными метками

Необходимость учитывать в модели данных элемент времени ставит перед проектировщиком ХД следующие задачи.

1. Выбор подхода к моделированию *темпоральных данных*.
2. Определение домена атрибута "время".
3. Определение структуры домена атрибута "время".
4. Определение атрибутов *временных меток*.
5. Нормализация временных зависимостей.
6. Выбор того, что моделировать — состояния или события?
7. Отображение *темпоральной модели данных* на исходную *статическую модель* данных.

Для моделирования *темпоральных данных* *предметной области* необходимо расширить модель введением в нее темпоральных (временных) сущностей или отношений. Такое расширение может быть сделано различными способами.

В настоящее время при создании *темпоральной модели данных* наиболее широко используются следующие подходы:

* накопление **моментальных снимков**, или **кумулятивных снимков** (*Cumulative* snapshots);
* поддержка истории изменений данных, или **непрерывной исторической модели** (Continuous history model), – *таблицы событий* и *состояний*.

Далее рассмотрим каждый из указанных подходов более подробно.

#### Модель, основанная на таблицах моментальных снимков

**Подход к моделированию** *темпоральных данных* **, основанный на накоплении моментальных снимков, состоит в сборе снимков фрагмента предметной области и накоплении таких снимков в различных фрагментах БД или другой БД как истории жизни данных предметной области**. Например, если снимки фиксируются в конце дня, то совокупность собранных за период снимков будут представлять ежедневное изменение картины данных предметной области, поддерживаемой в БД.

Таблица реляционной БД, представляющая моментальные снимки, называется **таблицей моментального снимка** (Snapshot Table). Рассмотрим пример такой таблицы ([табл. 7.1](https://intuit.ru/studies/courses/599/455/lecture/10165?page=1#table.7.1)). *Временные метки* в ней отсутствуют. Проектировщик может добавить *временную метку* со временем, определенным пользователем. Эта таблица представляет экземпляры сущности предметной области так, как они существуют в данный момент времени. Операция обновления таблицы реляционной БД является деструктивной – никакой истории изменения данных во времени не поддерживается.

В [табл. 7.6](https://intuit.ru/studies/courses/599/455/lecture/10165?page=2#table.7.6) показано содержание *таблицы моментального снимка* (Empl) до и после обновления.

|  |  |  |
| --- | --- | --- |
| Таблица 7.6. Изменение содержания *таблицы моментального снимка* | | |
| **До обновления** | | |
| **Сотрудник (Name)** | **Отдел (Dept)** | **Зарплата (Sal)** |
| Прохоров А.И. | ОВИР | 15000 |
| Соловьева М.Е. | ОВИР | 15000 |
| Амосова Е.С. | ОВИР | 6000 |
| **После обновления** | | |
| **Сотрудник (Name)** | **Отдел (Dept)** | **Зарплата (Sal)** |
| Прохоров А.И. | ОВИР | 15000 |
| Соловьева М.Е. | ОВИР | 15000 |
| Амосова Е.С. | ОВИР | 11500 |

Ниже приведены примеры операторов манипулирования данными SQL с *таблицей моментального снимка* Empl.

Удаление кортежа таблицы:

DELETE FROM Empl WHERE Name = 'Амосова Е.С.';

Добавление кортежа в таблицу:

INSERT INTO Empl VALUES ('Рыков А.С.','ОВИР','12000');

Обновление кортежа таблицы:

UPDATE Empl SET Dept = 'ОВИР', Sal ='11500' WHERE Name = 'Амосова Е.С.';

Метод кумулятивных снимков применяется и вне рамок моделирования *темпоральных данных*. Он прост и понятен, однако имеет ряд недостатков. Первый недостаток — это избыточность данных, которая возникает в результате периодической перезагрузки данных из источника данных. При этом объем ХД быстро растет. Чтобы устранить этот недостаток, часто применяют агрегирование данных с суммированием показателей или введение версий снимков.

Вторым недостатком такого подхода является возможная потеря информации. Этот недостаток связан с проблемой синхронизации изменений состояния данных в источниках данных и моментом фиксации снимка в ХД. Проблема отражения смены состояний данных предметной области в ХД является одной из самых сложных задач, которые должен решить проектировщик ХД. В рамках подхода, основанного на накоплении снимков, бывает достаточно трудно решить такую задачу. Например, если данные нужно фиксировать ежесекундно, то объем данных в ХД будет расти так быстро, что проект создания ХД станет непомерно дорогим. Такие ситуации могут встретиться в предметных областях, связанных с аудитом биржевых операций.

#### Модель, основанная на таблицах событий

Разработка непрерывной исторической модели ставит своей целью создание модели данных, которая отражает историю изменения данных в ХД. Это модель сложнее, чем модель кумулятивных снимков, в том числе и с точки зрения ее интерпретации. Несмотря на это, непрерывная историческая модель является более релевантной для анализа временных рядов параметров предметной области.

**Подход к моделированию** *темпоральных данных* **, основанный на фиксации событий предметной области, состоит в добавлении** *временной метки* **фиксации события (факта) как атрибута экземпляра сущности предметной области и отражении момента времени в таблице БД как истории жизни данных предметной области**. Например, учет оплаты счетов покупателем.

Таблица реляционной БД, представляющая события предметной области, называется ***таблицей событий*** (Event Table). Рассмотрим пример такой таблицы учета оплаты счетов клиентом ([табл. 7.7](https://intuit.ru/studies/courses/599/455/lecture/10165?page=3#table.7.7)). Событие для сущности предметной области "Счет" состоит в создании экземпляра сущности, содержащего данные о том, что в такое-то время покупатель оплатил счет на определенную сумму. *Временная метка* в этой таблице фиксирует время оплаты счета покупателем, т.е. представляет *время фиксации факта* оплаты счета. Проектировщик может добавить в такую таблицу *временную метку* со временем, определенным пользователем. Данные таблицы поддерживают, в частности, историю оплаты счетов конкретным покупателем. Обновление данных этой таблицы не производится, поскольку она предназначена для накопления данных (только операции добавления).

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 7.7. Оплата счетов покупателем | | | |
| **Номер счета** | **Покупатель** | **Сумма оплаты** | **Дата оплаты** |
| 1001 | Иванов А.И. | 1000 | 12.01.2009 |
| 1003 | Петров Е.С. | 25000 | 17.11.2008 |
| 1006 | Васильева Е.К. | 300 | 23.02.2009 |

*Таблицы событий* используются, как правило, для представления данных, по которым должен быть подведен итог, т.е. выполнена операция суммирования для различных периодов времени.

#### Модель, основанная на таблицах состояния

**Подход к моделированию** *темпоральных данных* **, основанный на фиксации состояний предметной области, состоит в добавлении** *временных меток* **для фиксации начала и завершения определенного состояния как атрибутов экземпляра сущности предметной области экземпляров сущности и отражении моментов времени начала и завершения определенного состояния сущности в таблице БД как истории жизни данных предметной области**.

Таблица реляционной БД, представляющая состояние объектов предметной области, называется **таблицей состояния** (State Table). Под состоянием понимаются объекты, которые существуют в определенный период времени. Пример таблицы состояния (Empl) приведен в [табл. 7.8](https://intuit.ru/studies/courses/599/455/lecture/10165?page=3#table.7.8). Кортежи таблицы – данные о зарплате сотрудников организации с указанием периода времени, когда установленная заплата выплачивается, т.е. состояние экземпляра сущности "Сотрудник" сохраняется. Пара *временных меток* "Время назначения" и "Время отмены" определяет период времени, в течение которого кортеж таблицы имеет смысл. Значение FOREVER равно наибольшему значению времени, принятому в системе.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблица 7.8. Содержание таблицы состояния Empl | | | | |
| **Сотрудник (Name)** | **Отдел (Dept)** | **Зарплата (Sal)** | **Время назначения (SDate)** | **Время отмены (EDate)** |
| Прохоров А.И. | ОВИР | 15000 | 01.06.2008 | FOREVER |
| Соловьева М.Е. | ОВИР | 15000 | 01.06.2008 | FOREVER |
| Амосова Е.С. | ОВИР | 10000 | 01.01.2008 | 30.05.2008 |
| Амосова Е.С. | ОВИР | 12000 | 01.06.2008 | FOREVER |

Ниже приведены операторы манипулирования данными SQL для таблицы состояния Empl.

Добавление кортежа в таблицу:

INSERT INTO Empl VALUES ('Рыков А.С.','ОВИР','12000',

'02.02.2009', FOREVER);

Обновление кортежа таблицы (обратите внимание на то, что обновление состоит из двух действий:

1. установка времени отмены на время, указанное в приказе по организации, и
2. добавление нового кортежа с временем назначения, равным времени, указанном в приказе по организации, и временем отмены, равным FOREVER ).

UPDATE Empl SET EDate="30.05.2008" WHERE Name = 'Амосова Е.С.'

AND Edate=FOREVER;

INSERT INTO Empl VALUES ('Амосова Е.С.','ОВИР','12000',

'01.06.20089', FOREVER);

Как правило, из таблиц состояния удаление кортежей таблицы выполняется после завершения обработки данных и после процесса их архивирования.

#### Так же, как и таблицы событий, таблицы состояния используются для хранения данных, по которым должен быть подведен итог, т.е. выполнена операция суммирования для различных периодов времени. Заметим, что в таблице событий суммирование проводится по одному периоду времени, в то время как в таблице состояния заданный период может перекрываться с периодами определенных сущностей, и, следовательно, при агрегации данных, возможно, придется проводить разбиение заданного периода на несколько периодов с учФормирование запросов к темпоральным данным

Разработка запросов с использованием *временных меток* требует от проектировщика особой внимательности и аккуратности. Рассмотрим формирование типовых предикатов с применением *временных меток*, а именно *предикатов сравнения* периодов времени. Пусть Р1 есть временной интервал "P1Start, Р1End", а P2 — временной интервал "P2Start, Р2End". Также введем условное обозначение VALID(T) для обозначения периода времени в кортеже таблицы. Заметим, что период времени не является *скалярной величиной* — это и создает определенные трудности в формировании *предикатов сравнения* периодов времени. Если сравниваются моментальные *временные метки*, то используется стандартный набор операторов сравнения скалярных величин SQL для типа данных DATA или TIMESTAMP.

Если P1Start < P2Start, то имеем *предикат сравнения* периодов времени "Меньше чем": P1 < P2 ([рис. 7.3](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.3)). Если P1End > P2End, то имеем *предикат сравнения* периодов времени "Больше чем": P1 > P2. Если P1Start = P2Start AND P1End = P2End, то периоды времени считаются эквивалентными. Этот тип предикатов применяется для упорядочивания периодов времени.

Предикат сравнения периодов времени "Меньше чем"

**Рис. 7.3.**Предикат сравнения периодов времени "Меньше чем"

Рассмотрим *предикат сравнения* периодов времени "Предшествует" (условное обозначение для примеров — *PRECEDES* ), когда один период времени предшествует другому ([рис. 7.4](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.4)), выполняется условие P1End < P2Start. Этот тип предикатов используется для разделения периодов времени.

Предикат сравнения периодов времени "Предшествует"

**Рис. 7.4.**Предикат сравнения периодов времени "Предшествует"

Для нахождения периодов времени, в которых происходит изменение значений факта, применяется *предикат сравнения* периодов времени "Встречаются" (условное обозначение для примеров — MEETS ). Если период времени P2, грубо говоря, цепляется своим началом за конец периода P1, то выполняется логическое условие (P1End = P2Start – 1) OR (P2End = P1Start – 1) (см. [рис. 7.5](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.5)). Примером использования такого предиката являются запросы типа "цена на товар растет или падает", "зарплата увеличена" и т.д.

[Предикат сравнения периодов времени "Встречаются"](https://intuit.ru/EDI/20_07_20_2/1595197216-9970/tutorial/632/objects/7/files/06_05.jpg)

**Рис. 7.5.**Предикат сравнения периодов времени "Встречаются"

Оператор SQL для выборки списка изменения цен на товар из таблицы Table1 может иметь вид:

SELECT B.No, B.Price. A.Price

FROM Table1 A, Table1 B

WHERE A.No=B.No AND A.Price < B.Price

AND VALID(A) PRECEDES VALID(B) AND VALID(A) MEETS VALID(B)

Комбинация временных предикатов означает, что сначала мы упорядочиваем периоды времени, а затем цепляем их друг за друга. Таким образом, получается непрерывная временная шкала периодов.

Для нахождения пересечений периодов времени используется *предикат сравнения* периодов времени "Перекрывает" (условное обозначение для примеров — OVERLAPS ). Если период времени P1 пересекается с периодом времени P2, то выполняется логическое условие (P1Start <= P2End) OR (P2Start <= P1End). Примером использования такого предиката является запрос типа "Найти всех сотрудников, которые работали в организации в период с 1 января 2008 года по 20 февраля 2009 года", например ([табл. 7.7](https://intuit.ru/studies/courses/599/455/lecture/10165?page=3#table.7.7)):

SELECT Name FROM Empl WHERE VALID(Empl) OVERLAPS [01.01.2008,

20.02.2009]

Для нахождения событий, которые происходили в течение определенного периода времени, используется *предикат сравнения* периодов времени "Содержится" (условное обозначение для примеров — CONTAINS ). Если период времени P1 содержится в периоде времени P2, то выполняется логическое условие (P1Start <= P2Start) OR (P1End <= P2End) (см. [рис. 7.6](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.6)).

Предикат сравнения периодов времени "Содержится"

**Рис. 7.6.**Предикат сравнения периодов времени "Содержится"

Различают несколько типов запросов к временным данным для таблиц состояния.

* **Запросы моментального снимка** (Snapshot query) – это запросы для построения выборки строк таблицы, действительных на определенный момент времени (дату).
* ***Неупорядоченные темпоральные запросы*** (Nonsequenced *temporal* query) – это запросы для построения выборки строк таблицы без учета *временных меток*, т.е. в полученной выборке строки по времени никак не соотносятся. В результате получается *таблица моментального снимка*.
* ***Упорядоченные темпоральные запросы*** (Sequenced *temporal* query) — это запросы для построения выборки строк таблицы на определенный период времени с использованием *предикатов сравнения* периодов времени, которые были рассмотрены выше. В результате получается *таблица состояния*, в которой *временная метка* для каждого кортежа удовлетворяет определенным временным ограничениям.

Приведем пример упорядоченного темпорального запроса. Рассмотрим таблицы БД Сотрудник (Empl) и Отдел (Department), содержание которых приведено в [табл. 7.9](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#table.7.9) и [6.10](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#table.7.10). Построим выборку ответа на вопрос "Кто руководил отделом Прохорова А.И.?" Результат приведен в [табл. 7.11](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#table.7.11).

SELECT Manager FROM Empl, Department

WHERE Department.Dept = Empl.Dept AND Empl.Name ='Прохоров А.И.'

AND VALID(Empl) OVERLAPS VALID(Department)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Таблица 7.9. Содержание таблицы состояния Empl | | | | | | | |
| **Сотрудник (Name)** | **Отдел (Dept)** | | **Зарплата (Sal)** | | **Время назначения (SDate)** | | **Время отмены (EDate)** |
| Прохоров А.И. | ОВИР | | 15000 | | 01.01.2008 | | 30.05.2008 |
| Прохоров А.И. | КИК | | 23000 | | 01.06.2008 | | FOREVER |
| Соловьева М.Е. | ОВИР | | 15000 | | 01.06.2008 | | FOREVER |
| Амосова Е.С. | ОВИР | | 10000 | | 01.01.2008 | | 30.05.2008 |
| Амосова Е.С. | ОВИР | | 12000 | | 01.06.2008 | | FOREVER |
| Таблица 7.10. Содержание таблицы состояния Department | | | | | | | |
| **Руководитель (Manager)** | | **Отдел (Dept)** | | **Время назначения (SDate)** | | **Время отмены (EDate)** | |
| Прохоров А.И. | | КИК | | 01.06.2008 | | FOREVER | |
| Соловьева М.Е. | | ОВИР | | 01.06.2008 | | FOREVER | |
| Амосова Е.С. | | ОВИР | | 01.01.2008 | | 30.05.2008 | |

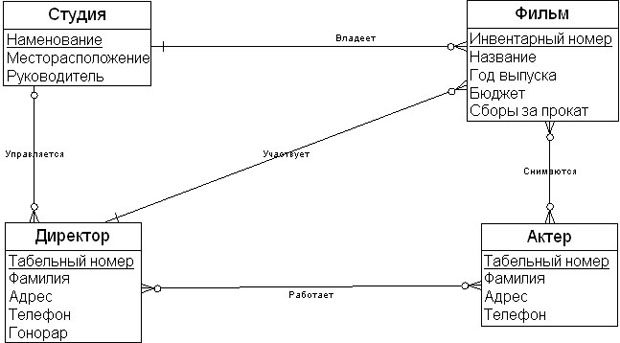
|  |  |  |
| --- | --- | --- |
| Таблица 7.11. Результат выполнения запроса | | |
| **Руководитель (Manager)** | **Время назначения (SDate)** | **Время отмены (EDate)** |
| Прохоров А.И. | 01.06.2008 | FOREVER |
| Амосова Е.С. | 01.01.2008 | 30.05.2008 |

Теперь давайте рассмотрим основные приемы моделирования *темпоральных данных*.

### Основные приемы моделирования темпоральных данных

#### Учет временных зависимостей предметной области

Рассмотрим приемы использования темпорального моделирования данных на практике. Рассмотрим пример БД о видеофильмах ("Фильм"), записанных на определенных студиях ("Студия") под управлением конкретных менеджеров ("Директор") с участием определенных актеров ("Актер"). Фрагмент логической структуры (ER-модели), связывающей указанные выше сущности, приведен на [рис. 7.7](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.7). Данная ER-модель является статической — она не учитывает никаких исторических аспектов существования студии или время работы определенного менеджера над определенной картиной.

[](https://intuit.ru/EDI/20_07_20_2/1595197216-9970/tutorial/632/objects/7/files/06_07.jpg)

**Рис. 7.7.**ER-модель БД о видеофильмах без учета временных зависимостей

Предположим, что перед проектировщиком ХД поставлена задача — построить историческую модель для приведенной ER-модели с помощью методов темпорального моделирования данных. Проектировщик должен рассмотреть временные зависимости сущностей предметной области и внести их в статическую ER-модель. **Временной зависимостью будем называть функциональную зависимость значений атрибутов сущностей предметной области от времени**.

Для учета временных зависимостей проектировщику необходимо добавить в приведенную статическую структуру *временные метки*. При этом ему следует решить следующие задачи:

* исходя из поставленной задачи, принять решение о том, поведение каких сущностей во времени будет учитываться в модели, а каких нет;
* определить, имеются ли сущности, которые могут содержать семантически содержательные для предметной области *временные метки*, и соответствующим образом их интерпретировать;
* добавить *временные метки* в сущности ER-модели. При этом следует принять решение о том, как будет моделироваться сущность – используя события или состояния. События используются для учета операций, которые изменяют данные в некоторые моменты времени. Состояния используются для представления состояния атрибутов сущности в течение заданного периода времени. В первом случае используются *моментные временные метки*, а во втором случае используются *интервальные временные метки* ;
* определить домены *временных меток* на имеющемся в СУБД типе "дата/время" и строго придерживаться этих определений доменов в дальнейшем.

При решении этих задач взаимоотношения между сущностями временно не принимаются во внимание.

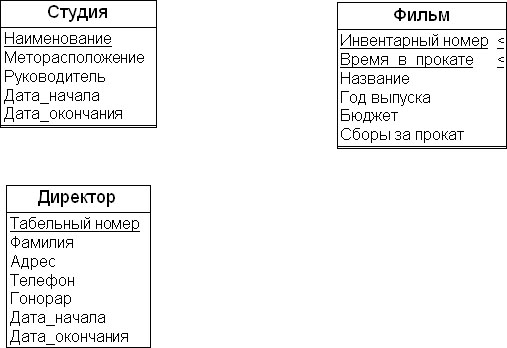
Предположим, что в нашем примере заказчику не интересна история взаимоотношения актеров и менеджеров (директоров) фильмов, а также история участия актеров в различных фильмах. Тогда для сущности, описывающей актеров, временное моделирование данных не будет выполняться.

Далее, пусть заказчику интересна история состояния студии или периоды занятости менеджеров фильмов. Тогда естественно моделировать временное поведение сущностей, описывающих студии и менеджеров, с помощью фиксации состояния их атрибутов.

Предположим, что заказчика интересует история существования фильма в прокате. Тогда было бы разумно выполнить временное моделирование сущности, описывающей фильм, с помощью фиксации событий, связанных с фильмом.

Например, одним из бизнес-приложений, необходимым для заказчика, будет "Прокат видеофильмов". Клиенты оплачивают пользование видеокассетой в течение определенного времени и дней. Приложение ведет учет предоставления клиентам таких услуг.

Результат темпорального моделирования на ER-модели показан на [рис. 7.8](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.8) ниже.



**Рис. 7.8.**Добавление временных меток в сущности статической ER-модели

Таким образом, основным приемом логического моделирования *темпоральных данных* предметной области является добавление *временных меток* в сущности предметной области. Добавление *временных меток* в сущности логической модели позволяет зафиксировать в модели поведение некоторых атрибутов сущности во времени. Такие атрибуты сущности образуют **временные зависимости атрибутов**.

#### Классы временных зависимостей атрибутов (нормализация модели)

После добавления *временных меток* в сущности логической модели одной из главных задач проектировщика становится определение атрибутов сущности, которые участвуют во временной зависимости. Одним из приемов решения такой задачи в рамках реляционной модели данных является нормализация временных зависимостей, или построение **классов временных зависимостей** (Volatility Classes).

***Классом временной зависимости*** **называется группа атрибутов сущности, которые совместно изменяют свои значения с течением времени**. Ответ на вопрос о том, сколько атрибутов должно попадать в класс, не является однозначным. Классы с небольшим кардинальным числом атрибутов невыгодны с практической точки зрения, а объединение в класс значительного числа независимо меняющихся во времени атрибутов приводит к увеличению избыточности хранения. Хорошим практическим критерием будет являться сложность сопровождения этих классов в ХД.

На практике проектировщики *темпоральных моделей данных* используют следующие два основных класса:

* **класс независимых от времени атрибутов** (Time-invariant Volatility Class);
* **класс зависимых от времени атрибутов** (Time-variant Volatility Class).

**Класс независимых от времени атрибутов образуют либо те атрибуты, которые не зависят от времени, либо те, для которых принято решение не хранить историю их временного поведения**. Этот класс всегда не пуст, поскольку предполагается, что ключ сущности составляют атрибуты, которые не зависят от временных отношений сущности в предметной области.

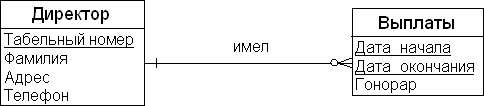
Атрибуты этого класса помещаются в новую сущность, которая представляет собой часть первоначальной сущности. Эта новая сущность будет являться как бы корнем в иерархии временных отношений со всеми зависимыми от времени сущностями, полученными из первоначальной сущности. С этой точки зрения история поведения сущности во времени может быть промоделирована с помощью зависимых сущностей.

Например, для сущности "Директор фильма" атрибуты *Табельный номер*, его *Фамилия*, *Адрес* и *Телефон* являются кандидатами для помещения в класс независимых от времени атрибутов. Понятно, что в этом случае предполагается, что директор не будет менять свою фамилию, и нас не интересует история его места жительства.

Для сущности "Фильм" такой класс может состоять из атрибутов *Инвентарный номер* фильма, *Название* фильма и *Год выпуска* фильма.

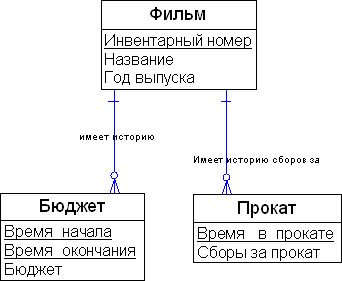
**Класс зависимых от времени атрибутов образуют те атрибуты сущности, которые меняются синхронно во времени**. Таких классов у сущности может быть несколько. Каждый такой класс должен быть вынесен в отдельную сущность.

Например, для сущности "Директор фильма" класс зависимых от времени атрибутов будут составлять *интервальная временная метка* и *Зарплата* директора. Такой класс будет единственным у первоначальной сущности. При этом будет моделироваться, как уже решил проектировщик, история состояния атрибута *Зарплата*. Ниже, на [рис. 7.9](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.9), показана историческая модель для сущности "Директор фильма".



**Рис. 7.9.**Темпоральная логическая модель для сущности "Директор фильма"

Для сущности "Фильм" могут быть идентифицированы два класса зависимых от времени атрибутов, поскольку значения атрибутов *Бюджет* фильма и *Сборы за прокат* имеют независимое друг от друга поведение во времени. Более того, поскольку значение атрибута *Бюджет* изменяется не слишком часто, для временного моделирования его поведения целесообразно использовать модель состояния и *интервальную временную метку*. Значение атрибута *Сборы за прокат* будет меняться при каждой записи данных в ХД, и следовательно, нужно отображать это событие с помощью *моментной временной метки*. Полученная логическая модель исторического поведения сущности "Фильм" приведена на [рис. 7.10](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.10) ниже.



**Рис. 7.10.**Темпоральная логическая модель для сущности "Фильм"

Проведя исследования исходной логической модели на предмет учета поведения сущностей во времени, проектировщик ХД должен ответить на вопрос, учтены ли все *темпоральные данные*, которые требуется хранить в ХД (если нет, то построить соответствующие сущности логической модели).

Решение этой задачи обычно связано с поиском сущностей для представления событий и операций хозяйственной деятельности, которые являются важными для решения задач анализа данных в ХД, но не поддерживаются в рамках исходной ER-модели данных подающих OLTP-систем. Эта задача является достаточно сложной и требует всестороннего изучения предметной области с точки зрения анализа данных.

Так, в нашем примере мы не рассматривали ряд вопросов, связанных с участием актеров в фильмах. Конечных пользователей может заинтересовать ряд вопросов, которые связаны с сущностью "Актер":

* С какими актерами чаще всего работает данный директор?
* Каких актеров данный директор не использует более одного раза?
* Как изменяется гонорар актеров по мере роста их популярности?
* На какие роли чаще приглашается данный актер?

Если на первые два вопроса пользователь может получить ответы на существующей структуре данных ER-модели (см. [рис. 7.7](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.7)), то для ответов на два других логическая модель должна быть дополнена информацией о гонорарах актера в фильмах и показателем, который измеряет популярность актера. Потребуются ли для представления такой информации дополнительные атрибуты или сущности, решает проектировщик ХД.

Таким образом, нормализация темпоральной логической модели основывается на выделении классов зависимых от времени атрибутов и вынесении их в отдельные сущности предметной области.

Теперь мы готовы приступить к построению логической темпоральной модели предметной области ХД о видеофильмах.

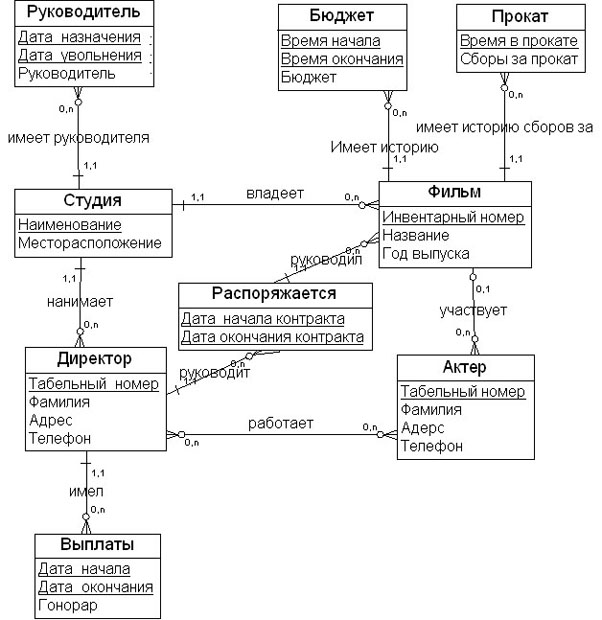
#### Построение логической темпоральной модели данных

После того, как проектировщик построит темпоральные модели сущностей исходной ER-модели, ему предстоит собрать все темпоральные модели сущностей в одной логической модели данных и зафиксировать все необходимые взаимосвязи.

Для каждой взаимосвязи исходной логической модели проектировщик ХД должен исследовать вопрос: нужно ли хранить историю взаимосвязи в ХД? Если историю изменений взаимосвязи не нужно сохранять в ХД, то взаимосвязь просто переносится в построенную временную модель сущностей как взаимосвязь между сущностями, представляющими классы независимых от времени атрибутов.

Если историю изменений взаимосвязи нужно сохранять в ХД, то такую взаимосвязь нужно дополнить атрибутами *временных меток*, представив ее в виде сущности. В реляционной модели данных взаимосвязь между сущностями моделируется с помощью отношений и, следовательно, должна быть представлена с помощью сущностей.

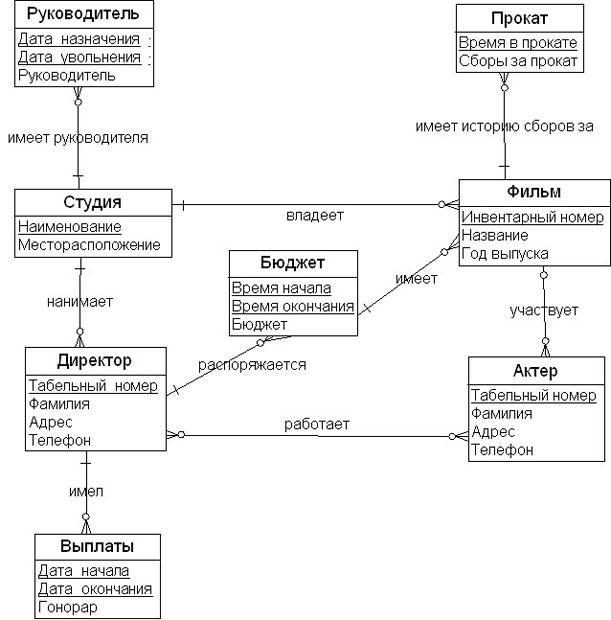
Для рассматриваемого нами примера *темпоральная модель данных* может быть такой, как на [рис. 7.11](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.11). Как видно из рисунка, четыре исходные взаимосвязи (владеет, нанимает, работает и имеет) перенесены во временную модель данных из ER-модели без изменений, поскольку информацию об их временном поведении было решено не сохранять.

[](https://intuit.ru/EDI/20_07_20_2/1595197216-9970/tutorial/632/objects/7/files/06_11.jpg)

**Рис. 7.11.**Темпоральная модель данных БД о видеофильмах

В сущности "Студия" решено хранить историю смены ее руководителей, поэтому в модель была добавлена сущность "Руководитель студии". Темпоральные модели сущностей "Фильм" и "Директор" были построены ранее, и для них зафиксированы взаимосвязи с другими сущностями модели. А вот взаимосвязь "руководит" исходной ER-модели представлена в виде сущности "Распоряжается" (фильмом), поскольку решено сохранять историю работы менеджеров над фильмами.

Проектировщик ХД может далее исследовать полученную модель, для того чтобы уменьшить ее сложность, например, сократив число сущностей в модели. В нашем случае нетрудно заметить, что период времени, когда менеджер работает над фильмом, совпадает с периодом времени, когда на фильм выделены деньги. Следовательно, сущности "Распоряжается" и "Бюджет" могут быть объединены в одну сущность "Бюджет", как показано на [рис. 7.12](https://intuit.ru/studies/courses/599/455/lecture/10165?page=4#image.7.12). Такой прием называется **группировкой классов атрибутов, зависимых от времени**. Заметим, что использование такого приема приводит к увеличению избыточности данных в модели.

[](https://intuit.ru/EDI/20_07_20_2/1595197216-9970/tutorial/632/objects/7/files/06_12.jpg)

**Рис. 7.12.**Преобразование полученной темпоральной модели данных БД о видеофильмах

Таким образом, некоторые временные ограничения, такие, которые содержат временной аспект, представляются в модели в виде сущностей-взаимосвязей. Однако существуют временные ограничения, которые выражаются в виде бизнес-правил. Такие ограничения трудно моделировать в рамках расширенной ER-модели, которая отображается в реляционную модель при помощи стандартных правил проектирования. Например, может существовать бизнес-правило: два менеджера не могут одновременно работать над одним фильмом. Обычно такие бизнес-правила формулируются при помощи описательного текста и используются разработчиками приложений, т.е. моделируются кодом приложения БД.

Следует отметить также, что временные ограничения следует заносить в *словарь метаданных*, поскольку они являются важной информацией для конечного пользователя.

Таким образом, логическая темпоральная модель предметной области ХД о видеофильмах построена.

### Резюме

Рассмотренный в настоящей лекции подход к моделированию *темпоральных данных* является расширением модели "*сущность-связь*" за счет учета временных зависимостей определенных атрибутов сущностей *предметной области*.

При таком подходе проектировщик выполняет следующую последовательность действий.

* Разрабатывает модель "сущность-связь" для выделенной предметной области (ER-диаграммы, как правило, в третьей нормальной форме) без учета временных зависимостей атрибутов модели.
* Исходя из бизнес-требований определяет атрибуты модели, которые зависят от времени и будут учитываться в разрабатываемой модели (т.е. принимает решение о выделении совокупности *темпоральных данных* модели).
* Выполняет учет временных зависимостей атрибутов посредством введения в сущности *временных меток*.
* Нормализует полученные темпоральные сущности посредством выделения классов зависимых от времени атрибутов и вынесения их в отдельные сущности.
* Принимает решение о представлении взаимосвязей предметной области (в том числе и зависящих от времени) в виде сущностей предметной области.
* Собирает полученные сущности в единую ER-модель. Эта модель, как правило, нормализована.
* Принимает решение о денормализации модели, исходя из требований производительности запросов или из обоснованных соображений по упрощению модели (например, наглядность).

В результате применения темпорального моделирования проектировщик создает логическую модель *предметной области* с учетом временных зависимостей данных, независимую от программно-аппаратной реализации. Этот метод используется как при моделировании предметных областей OLTP-систем, так и при моделировании предметных областей BI-систем.

*Знание* техники построения логических *темпоральных моделей данных* особенно важно при проектировании ХД, поскольку одной из целей создания последних является исследование временных зависимостей данных организации.