

## КОНСПЕКТ ЗАНЯТИЯ № 2 ВТОРОЙ НЕДЕЛИ КУРСА «УПРАВЛЕНИЕ ДАННЫМИ»

### 3.2. СЕТЕВАЯ МОДЕЛЬ ДАННЫХ

Стандарт сетевой модели данных был опубликован в отчете организации CODASYL (*от англ. CO*nference on *DA*tа *SY*stems Languages), в 1971 году. Также как и иерархическая модель, сетевая модель относится к разряду теоретико-графовых, но она позволяет строить структуры данных, описываемые графом более общего вида, чем предполагает иерархическая модель. Важное влияние на создание и развитие сетевой модели данных оказал проект по разработке СУБД IDS (Integrated Data Store) осуществлявшийся в 1960-х в корпорации General Electric под руководством Чарльза Бахмана. Другой известный проект – СУБД IDMS (Integrated Database Management System), разработка которой начиная с 1970-х велась различными компаниями, в частности, в начале 1980-х продукт разрабатывался Cullinane Database Systems с участием того же Чарльза Бахмана. В настоящее время эта СУБД для мейнфреймов продолжает развиваться и права на нее принадлежат CA Technologies.

Базовые структуры данных сетевой модели: элемент данных, агрегат данных, запись (или группа), набор (групповое отношение), база данных.

*Элемент данных* (или просто «элемент») – минимальная именованная единица данных, доступная пользователю с помощью СУБД.

*Агрегат данных* – именованная совокупность элементов или других агрегатов данных. Разница между элементом и агрегатом может быть проиллюстрирована следующим примером. Пусть, в базу данных вносятся адреса. Если разработчик рассматривает адрес как единое целое (и соответствующим образом проектирует БД), то адрес – это элемент данных. Если же необходимо разделить адрес на части («страна»-«город»-«улица»-«номер дома»-«номер квартиры»), то адрес

уже будет выступать как агрегат, состоящий из соответствующих элементов. При этом пользователь может запросить из БД как отдельно город или номер дома, так и адрес целиком, так как агрегат – это тоже именованный объект.

Различают агрегаты типа «вектор» и «повторяющаяся группа» [4,8]. Агрегат, состоящий из простых элементов данных, называется *вектором*. Примером агрегата типа вектор может служить упомянутый выше адрес, представимый как совокупность элементов. Агрегат, повторяющийся компонент которого представлен совокупностью данных, называется *повторяющейся группой*. В повторяющуюся группу могут входить элементы данных и другие агрегаты. В качестве примера здесь можно привести агрегат «Оценки студента», в котором будет сохраняться название предмета и оценка, причем столько раз, сколько экзаменов сдаст студент.

*Запись* – это агрегат, который не входит в состав никакого другого агрегата, обычно описывает некоторый объект реального мира и составляет основную единицу обработки БД (записи запоминаются, извлекаются, удаляются).

*Набор* в сетевой модели является иерархическим отношением между двумя типами записей, т. е. экземпляр подчиненной записи не может быть участником двух экземпляров набора одного типа. В сетевой модели один и тот же тип записи может участвовать в нескольких наборах. В частности, для любых двух типов записей может быть задано любое количество наборов, которые их связывают. Наличие подобных возможностей позволяет моделировать отношение объектов типа «многие-ко-многим», что выгодно отличает сетевую модель данных от иерархической.

В то же время, ни в иерархической, ни в сетевой модели один и тот же тип записи не может быть одновременно и владельцем, и членом группового отношения (набора).

На рис. 3.2 с помощью диаграмм Бахмана проиллюстрировано различие между групповыми отношениями иерархической и сетевой моделях данных.

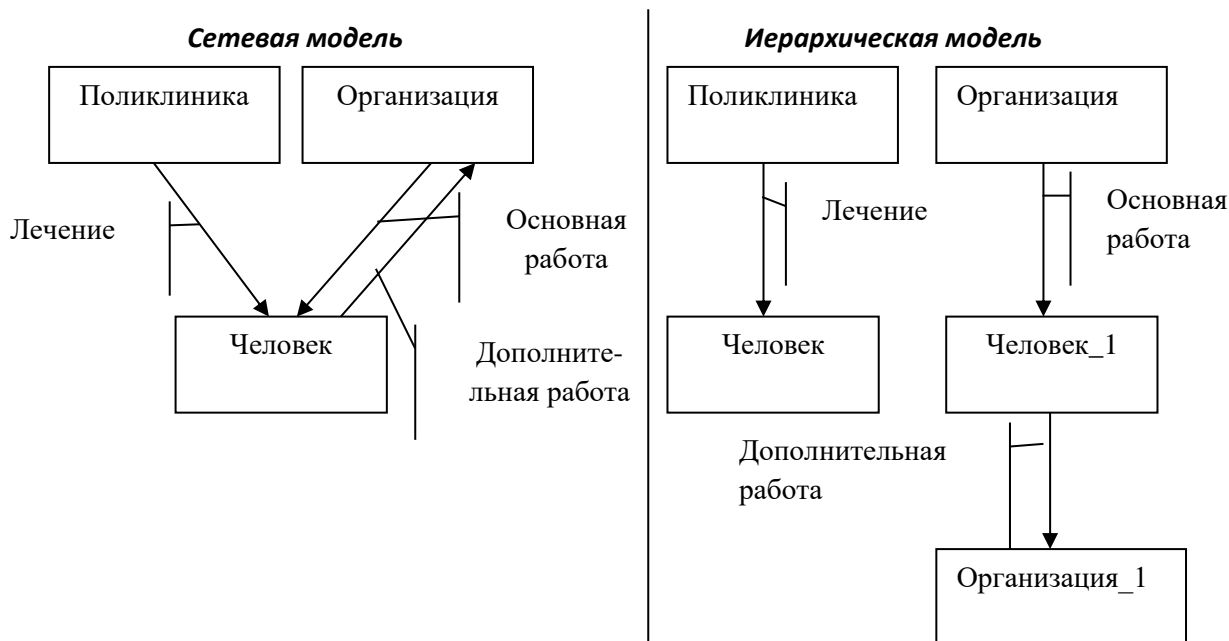


Рис. 3.2 Особенности групповых отношений в сетевой и иерархической моделях данных

На диаграмме отображены отношения между типами записей «Человек», «Организация», «Поликлиника». В рамках сетевой модели можно описать, что между типами записей «человек» и «организация» связь многие-ко-многим: один человек может работать в нескольких организациях и в одной организации может работать несколько человек. В иерархической модели в аналогичном случае придется ввести два типа записей «Организация» и «Организация\_1», и частично продублировать значения.

В сетевой модели отдельно выделяется так называемый *сингулярный набор*, владельцем которого формально определена вся система. Сингулярный набор изображается в виде входящей стрелки, которая имеет имя набора, имя члена набора, но у которой не определен тип записи «владелец набора». Сингулярные наборы позволяют обеспечить произвольный доступ к некоторому типу записи.

Каждый тип набора характеризуется следующими свойствами [9]:

- способом упорядочения подчиненных записей;
- режимом включения подчиненных записей;

- режимом исключения подчиненных записей.

Каждый экземпляр набора можно рассматривать как список записей-членов, поставленных в соответствие некоторой записи-владельцу. Записи-члены могут быть упорядочены в таком списке следующими способами:

- произвольным;
- хронологическим – в последовательности их поступления (очередь);
- обратнoхронологическим (стек);
- сортированным – в подчиненной записи выделяется ключ упорядочения, а место новой записи в списке определяется значением этого ключа.

Режим включения записей в набор может быть автоматическим или ручным. При *автоматическом включении* подчиненная запись включается в набор одновременно с ее запоминанием в БД. *Ручное включение* позволяет запомнить в БД подчиненную запись и не включать ее немедленно в экземпляр набора. Эта операция выполняется позднее по команде пользователя.

Выделяется три класса членства подчиненных записей в наборах: фиксированное, обязательное и необязательное. Тип членства определяет и режим исключения записи из набора.

При *фиксированном членстве* подчиненная запись жестко закрепляется за записью-владельцем. Она не может существовать без этого владельца. Подчиненную запись можно исключить из экземпляра группового отношения, только удалив ее из БД. При удалении владельца автоматически удаляются все подчиненные записи.

*Обязательное членство* означает, что каждая подчиненная запись, будучи однажды включена в групповое отношение, впредь всегда будет связана с некоторой записью-владельцем. Допускается переключение подчиненной записи к другому владельцу, но не допускается ее существование без владельца. Для успешного удаления записи-владельца необходимо, чтобы она не имела подчиненных записей с обязательным членством.

*Необязательное членство* позволяет исключить подчиненную запись из экземпляра группового отношения, но сохранить ее в БД, не прикрепляя к другому владельцу. При удалении записи-владельца подчиненные записи сохраняются в БД, не участвуя более в соответствующем наборе.

В общем случае сетевая база данных представляет собой совокупность взаимосвязанных наборов, которые образуют на концептуальном уровне некоторый граф [4].

Сетевая модель определяет следующие операции над записями:

- *запомнить* (Store) – позволяет внести в БД новую запись;
- *включить в набор* (Connect) – связывает существующую подчиненную запись с записью-владельцем;
- *обновить* (Modify) – позволяет изменить значения элементов существующих в БД записей;
- *найти* (Find) – всегда можно найти запись по значению первичного ключа. Также можно найти запись, используя групповые отношения, в которых она участвует;
- *извлечь* (Get) – позволяет извлечь запись (после того, как она найдена);
- *удалить* (Erase) – удаляет текущий экземпляр записи;
- *исключить из набора* (Disconnect) – исключение текущей записи из текущего экземпляра набора.

## БИБЛИОГРАФИЧЕСКИЙ СПИСОК

4. Карпова Т.С. Базы данных: модели, разработка, реализация. – СПб.: Питер, 2002 – 304 с.

8. Советов Б.Я. Базы данных: Теория и практика. Учеб. для втузов / Б.Я. Советов, В.В. Цехановский, В.Д. Чертовской. –2-е изд., стер. – М.: Высшая школа, 2007.– 463 с.

9. Бойко В.В., Савинков В.М. Проектирование баз данных информационных систем. – М: «Финансы и статистика», 1989. – 350 с.