### 2. ТЕХНОЛОГИИ СОЗДАНИЯ РАСПРЕДЕЛЕННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Построение современных распределенных информационных систем сегодня на прямую связано с реаляционными и объектно-ориентированными СУБД, которые в последнее время утвердились как основные средства для обработки данных в информационных системах различного масштаба - от больших приложений обработки транзакций в банковских системах до персональных систем на РС. В настоящее время существует множество систем управления базами данных (СУБД) и других программ выполняющих сходные функции. Инструментальные средства Oracle - одни из лучших и наиболее мощных имеющихся инструментов разработки профессионального класса.

#### 2.1 Базы данных и их сравнительные характеристики.

#### 2.1.1 Классификация моделей построения баз данных

В зависимости от архитектуры СУБД делятся на *локальные и распределенные* СУБД. Все части локальной СУБД размещаются на одном компьютере, а распределенной на нескольких. За несколько десятилетий последовательно появлялись системы (СУБД), основанные на трех базовых моделях данных: иерархической, сетевой и реляционной. Основные определения теории баз знаний и баз данных представлены в таблице 1.

База данных (БД) ­ Базами данных называют электронные хранилища информации, доступ к которым осуществляется с помощью одного или нескольких компьютеров.

Системы управления базами данных (СУБД) ­ это программные средства для создания, наполнения, обновления и удаления баз данных.

База знаний ­ Базы знаний это хранилища знаний, представленных в формализованном виде.

Система управления базами знаний СУБЗ ­ это программные средства для создания, наполнения, обновления и удаления баз знаний.

Виды знаний:

* Процедурные ­ Знания, отвечающие на вопрос "Как решать поставленную задачу?"
* Декларативные ­ Знания, не содержащие в явном виде процедуры решения задач.
* Каузальные ­ Знания о причинно-следственных связях между объектами предметной области
* Неточные ­ Знания отличающиеся неполнотой или противоречивостью.

Парадигмы решения задач:

* В СУБД ­ Данные + Алгоритм = Программа решения задачи.
* В СУБЗ ­ Знания + Стратегия вывода = Решение проблемы.

Модели знаний:

* Продукционная ­ Знания представленные в формате "ЕСЛИ-ТО".
* Фреймовая ­ Знания представленные в виде набора взаимосвязанный фреймов.
* Семантическая сеть ­ Граф, вершины которого соответствуют объектам или понятиям, а дуги определяют отношения между вершинами.
* Фрейм ­ Структурированное описание объекта предметной области состоящее из наименования объекта (имя фрейма), атрибутов объекта (свойств, характеристик) - слоты фрейма.
* Фрейм прототип ­ Это фрейм у которого значения слотов не определены.
* Конкретный фрейм ­ Это фрейм прототип с конкретными значениями.

Enterprise JavaBeans ­ Стандарт для создания средствами языка Java пригодных для многократного использования компонентов, из которых формируются прикладные программы. Компоненты Enterprise JavaBeans облегчают разработку программ, обеспечивающих доступ к хранимой в базе данных информации.

Распараллеливание обработки запроса (Intraquery parallelism) ­ Использование нескольких ЦП для обработки одного запроса.

Параллельная обработка запросов (interquery parallelism) подразумевает параллельную обработку нескольких запросов (на разных ЦП).

Уровень изоляции (Isolation level).Установочный параметр БД, определяющий, в какой степени одновременно обратившиеся к базе данных пользователи могут оказывать влияние на работу друг друга. Как правило, используются три уровня изоляции: *завершение чтения* (read committed), характеризуется большим количеством одновременно обслуживаемых пользователей и низким уровнем изоляции каждого из них); *в установленном порядке* (serializable), небольшое число одновременно обслуживаемых пользователей, высокая степень изоляции *и повторяющееся чтение* (repeatable read), сочетание двух первых уровней.

Технология СОМ ­ COM - Component Object Model - Компонентная модель объектов, предложена корпорацией Микрософт.

|  |  |
| --- | --- |
| *Технология CORBA* | CORBA - Common Object Require Broker Architecture - архитектура с брокером требуемых общих объектов, разработана независимой группой OMG. |
| *JDBC (Java Database Connectivity).* | Интерфейс взаимодействия с базами данных на языке Java. Этот стандарт, разработанный фирмой Sun Microsystems, определяет способы доступа Java-приложений к данным БД. |
| *ODBC (Open Database Connectivity).* | Открытый интерфейс взаимодействия с базами данных. Предложенный корпорацией Microsoft стандарт, регулирующий доступ Windows -приложений к базам данных. Стандарт ODBC постепенно заменяется спецификацией OLE DB. |
| *OLAP (Online analytical processing).* | Оперативный анализ данных. Этот метод обработки применяется с целью ускорения обработки запросов и предусматривает предварительный расчет часто запрашиваемых данных (например, сумм или значений счетчика). |
| *OLE DB (Object Linking and Embedding Database).* | OLE для баз данных. Новый стандарт Microsoft, регулирующий доступ приложений к базам данных. Имеет расширения для серверов OLAP и предусматривает применение специальных средств обработки мультимедийных данных. |
|  | *Дополнения к табл.2* |
| *Операция соединения (Join).* | Процесс, позволяющий объединять данные из двух таблиц посредством сопоставления содержимого двух аналогичных столбцов. |
| *SQL (Structured query language).* | Язык структурированных запросов, язык S0L. Является принятым в отрасли стандартом для выполнения операций вставки, обновления, удаления и выборки данных из реляционных БД. |
| *Хранимая процедура (Stored procedure).* | Программа, которая выполняется внутри базы данных и может предпринимать сложные действия на основе информации, задаваемой пользователем. Поскольку хранимые процедуры выполняются непосредственно на сервере базы данных, обеспечивается более высокое быстродействие, нежели при выполнении тех же операций средствами клиента БД. |
| *Транзакция (Transaction).* | Совокупность операций базы данных, выполнение которых не может быть прервано. Для того чтобы изменения, внесенные в БД в ходе выполнения любой из входящих в транзакцию операций, были зафиксированы в базе данных, все операции должны завершиться успешно. Все базы данных, представленные в нашем обзоре, позволяют использовать транзакции, тогда как БД для настольных систем, например Visual dBase фирмы Inprise или Microsoft Access, не предусматривают применения механизма транзакций. |
| *Триггер (Trigger).* | Программа базы данных, вызываемая всякий раз при вставке, изменении или удалении строки таблицы. Триггеры обеспечивают проверку любых изменений на корректность, прежде чем эти изменения будут приняты |

#### 

#### 2.1.1.1 Иерархическая модель

Первые иерархические и сетевые СУБД были созданы в начале 60-х годов. Причиной послужила необходимость управления миллионами записей (связанных друг с другом иерархическим образом), например при информационной поддержке лунного проекта Аполлон. Среди реализуемых на практике СУБД этого типа преобладает система *IMS* (Information Management System компании IBM) (На данный момент это самая распространенная СУБД из всех данного типа). Применяются и другие иерархические системы: *TDMS (*Time*-*Shared Date Management System) компании Development Corporation; *Mark IV* Multi - Access Retrieval System компании Control Data Corporation; *System - 2000* разработки SAS-Institute.

Отношения в иерархической модели данных организованы в виде совокупностей деревьев, где *дерево* - структура данных, в которой тип сегмента потомка связан только с одним типом сегмента предка. Графически: *Предок -* точка на конце стрелки, а *Потомок -* точка на острие стрелки. В базах данных определено, что точки - это типы записей, а стрелки представляют отношения один - к - одному или один - ко - многим.

К ограничения иерархической модели данных можно отнести:

1. Отсутствует явное разделение логических и физических характеристик модели;
2. Для представления неиерархических отношений данных требуются дополнительные манипуляции;
3. Непредвиденные запросы могут требовать реорганизации базы данных.

#### 

#### 2.1.1.2 Сетевая модель.

*Сети* - естественный способ представления отношений между объектами. Они широко применяются в математике, исследованиях операций, химии, физике, социологии и других областях знаний. Сети обычно могут быть представлены математической структурой, которая называется *направленным графом.* Направленный граф имеет простую структуру. Он состоит из точек или *узлов,*соединенных стрелками или ребрами. В контексте моделей данных узлы можно представлять как типы записей данных, а ребра представляют отношения один-к -одному или один-ко-многим. Структура графа делает возможными простые представления иерархических отношений (таких, как генеалогические данные) .

*Сетевая модель данных* - это представление данных сетевыми структурами типов записей и связанных отношениями мощности один-к-одному или один-ко-многим. В конце 60-х конференция по языкам систем данных (Conference on Data Systems Languages, CODASYL) поручила подгруппе, названной Database Task Group (DTBG), разработать стандарты систем управления базами данных. На DTBG оказывала сильное влияние архитектура, использованная в одной из самых первых СУБД, Iategrated Data Store (IDS), созданной ранее компанией General Electric.Это привело к тому, что была рекомендована сетевая модель.

Документы Database Task Group (DTBG) (группа для разработки стандартов систем управления базами данных) от 1971 года остается основной формулировкой сетевой модели, на него ссылаются как на модель CODASYL DTBG. Она послужила основой для разработки сетевых систем управления базами данных нескольких производителей. IDS (Honeywell) и IDMS (Computer Associates) - две наиболее известных коммерческих реализации. В сетевой модели существует две основные структуры данных: типы записей и наборы:

* *Тип записей.* Совокупность логически связанных элементов данных.
* *Набор.* В модели DTBG отношение один-ко-многим между двумя типами записей.
* *Простая сеть.* Структура данных, в которой все бинарные отношения имеют мощность один-ко-многим.
* *Сложная сеть.* Структура данных, в которой одно или несколько бинарных отношений имеют мощность многие-ко-многим.
* *Тип записи связи*. Формальная запись, созданная для того, чтобы преобразовать сложную сеть в эквивалентную ей простую сеть.

В модели DBTG возможны только простые сети, в которых все отношения имеют мощность один-к-одному или один-ко-многим. Сложные сети, включающие одно или несколько отношений многие-ко-многим, не могут быть напрямую реализованы в модели DBTG. Следствием возможности создания искусственных формальных записей является необходимость дополнительного объема памяти и обработки, однако при этом модель данных имеет простую сетевую форму и удовлетворяет требованиям DBTG.

#### 2.1.1.3 Реляционная модель.

В 1970-1971 годах Е.Ф.Кодд опубликовал две статьи, в которых ввел реляционную модель данных и реляционные языки обработки данных - реляционную алгебру и реляционное исчисление.

* *Реляционная алгебра* Процедурный язык обработки реляционных таблиц.
* *Реляционное исчисление* Непроцедурный язык создания запросов.

Все существующие к тому времени подходы к связыванию записей из разных файлов использовали физические указатели или адреса на диске. В своей работе Кодд продемонстрировал, что такие базы данных существенно ограничивают число типов манипуляций данными. Более того, они очень чувствительны к изменениям в физическом окружении. Когда в компьютерной системе устанавливался новый накопитель или изменялись адреса хранения данных, требовалось дополнительное преобразование файлов. Если к формату записи в файле добавлялись новые поля, то физические адреса всех записей файла изменялись. То есть такие базы данных не позволяли манипулировать данными так, как это позволяла бы логическая структура. Все эти проблемы преодолела **реляционная модель, основанная на логических отношениях данных.**

Существует два подхода к проектированию реляционной базы данных.

* *Первый подход* заключается в том, что на этапе концептуального проектирования создается не концептуальная модель данных, а непосредственно реляционная схема базы данных, состоящая из определений реляционных таблиц, подвергающихся нормализации.
* *Второй подход* основан на механическом преобразовании функциональной модели, созданной ранее, в нормализованную реляционную модель. Этот подход чаще всего используется при проектировании больших, сложных схем баз данных, необходимых для корпоративных информационных систем.

Табл.1. Основные определения реляционных СУБД

|  |  |  |
| --- | --- | --- |
| № | Термин | Определение |
| 1 | *Реляционная модель данных* | Организует и представляет данные в виде таблиц или реляций. |
| 2 | *Реляционная база данных (РБД, RDBMS).* | База данных, построенная на реляционной модели. |
| 2 | *Реляция (таблица-элементарная информационная единица)* | Двумерная таблица, содержащая строки и столбцы данных. |
| 4 | *Степень реляции.* | Количество атрибутов реляции. При том необходимо помнить, что никакие два атрибута реляции не могут иметь одинаковых имен. |
|  | *Кортежи* | Строки реляции (таблицы), соответствуют объекта, конкретному событию или явлению. |
|  | *Атрибуты* | Столбцы таблицы, характеризующие признаки, параметры объекта, события, явления. |
|  | *Область атрибута* | Набор всех возможных значений, которые могут принимать атрибуты. Если в процессе работы возникает ситуация, что атрибут неприменим или значения одного или нескольких атрибутовстроки пока неизвестны, то строка запишется в базуданных с пустыми значениямиэтих атрибутов (NULL строка). |
|  | *Пустое значение* | Значение, приписываемое атрибуту в кортеже, если атрибут неприменим или его значение неизвестно |
|  | *Ключ* | Любой набор атрибутов, однозначно определяющий каждый кортеж реляционной таблицы. |
|  | *Ключ реляции* | Ключ также можно описать как минимальное множество атрибутов, однозначно определяющих (или функционально определяющих)каждое значение атрибута в кортеже. |
|  | *Составной* ключ | Ключ содержащий два или более атрибута. |
|  | *Потенциальный ключ* | В любой данной реляционной таблице может оказаться более одного набора атрибутов. Обычно в качестве *первичного ключа* выбирают потенциальный ключ, которым проще всего пользоваться при повседневной работе по вводу данных. |
|  | *Первичный ключ.* | Поле или набор полей, однозначно идентифицирующий запись. |
|  | *Внешний ключ.* | Набор атрибутов одной таблицы, являющийся ключом другой (или той же самой) таблицы; используется для определения логических связей между таблицами. Атрибуты внешнего ключа не обязательно должны иметь те же имена, что и атрибуты ключа, которым они соответствуют. |
|  | *Рекурсивный внешний ключ.* | Внешний ключ, ссылающийся на свою собственную реляционную таблицу. |
|  | *Родительская реляция (таблица)* | Таблица, поля которой входят в другую таблицу. |
|  | *Дочерняя реляция (таблица)* | Таблица, поля которой используют информацию из полей другой таблицы, являющейся по отношению к данной родительской. |
|  | *Отношение один-к-одному* | Когда одной записи в родительской таблицы соответствует одна запись в дочерней таблице |
|  | *Отношение один-ко-многим* | Когда одной записи в родительской таблицы соответствует несколько записей в дочерней таблице |
|  | *Отношение многие-ко-многим* | Когда многим записям в родительской таблицы соответствуют несколько записей в дочерней таблице |
|  | *Рекурсивное отношение*. | Отношение, связывающее объектное множество с ним самим. |
|  | *View (Представления)* | Информационная единица РБД (по структуре аналогичная таблице), записи которой сформированы в результате выполнения запросов к другим таблицам. |
|  | *Ссылочная целлостность* | Адекватное воспроизведение записей в ссылочных полях таблиц. |
|  | *Триггер* | Средство обеспечения ссылочной целостности на основе механизма каскадных изменений. |
|  | *Индекс* | Механизмы быстрого доступа к хранящимся в таблицах данных путем их предварительной сортировки. |
|  | *Транзакция* | Такое воздействие на СУБД, которое переводит ее из одного целостного состояния в другое. |