#### 2.3.1. Исследование моделей информационного представления данных в современных СУБД.

Исследование различным методов ис средств представления и управления данными в информационных системах проведем на примере интерактивной базы данных патентного обеспечения (ПО) конструкторско-технологического проектирования (КТП). Патент - это документ, свидетельствующий о праве изобретателя на его изобретение. Для стандартизации и облегчения поиска информации были введены различные классификации патентов: (Национальная Классификация Патентов (НКИ), Универсальная Десятичная Классификация (УДК), Международная Классификация Изобретений (МКИ)). Все эти классификации призваны служить инструментом для упорядоченного хранения патентных документов, что облегчает доступ к содержащейся в них информации, быть основой для избирательного распределения информации среди потребителей патентной информации и для получения систематических данных в области промышленного соответствия, что в свою очередь, определяет уровень развития различных областей техники.

Классификации патентов имеют сложную структуру, и для поиска необходимой информации может потребоваться много времени. Возможна организация поиска по всем имеющимся классификациям изобретений, но пока, в качестве примера, мы рассмотрим только Международную Классификацию Изобретений, которая являясь средством для единообразного в международном масштабе классифицирования патентных документов, представляет собой эффективный инструмент для патентных ведомств и других потребителей, осуществляющих поиск патентных документов с целью установления новизны и оценки вклада изобретения в заявленное техническое решение (включая оценку технической прогрессивности и полезности или результата).

Международная Классификация Изобретений (МКИ) имеет иерархическую структуру (представленную на рис.1) и состоит из следующих отделов: 1 - Раздел, 2 - Класс, 3 - Подкласс, 4 - Группа, 5 - Подгруппа. Иерархическая структура классификации МКИ представлена на рис.31.

**Логическая модель**

Переход от функциональной модели к логической осуществляется с помощью реляционных методов, при этом иерархическая структура функциональной модели реализуется с использованием отношений один - ко -многим и рекурсивных рис. 27. Реализацией данной логической модели является совокупностью таблиц, объединенных в единый модуль - патентную информационную базу данных ( PIB - Patent Information dateBase).

Ядром логической модели является таблица PIB\_MKI (МКИ), связывающая таблицы PIB\_PART (Раздел), PIB\_CLASS (Класс), PIB\_SUBCLASS (Подкласс), PIB\_GROUP (Группа), PIB\_SUB-GROUP (Подгруппа) в единую структуру, определяющую реализацию Международной Классификации Изобретений (МКИ) винформационной системе патентного обеспечения технологического проектирования. Таблица PIB\_MKI (МКИ) в свою очередь связана с таблицей PIB\_PATENT (Патент), отвечающей за связь с таблицами PIB\_GRATHDOC (Графические документы) и PIB\_UDK (УДК).Таблица PIB\_UDK (УДК) реализует Универсальную Десятичную Классификацию (УДК). Структура таблиц модуля PIB представлена в таблице1.

Таблица 1. Информационно-логическая Структура модуля Международной Классификации Изобретений.

|  |  |  |
| --- | --- | --- |
| Имя таблицы | Имя поля | Функц. Назначение |
| PIB\_PART | PART\_NNN | Уникальный идентификатор |
|  | PART\_INDEX | Индекс раздела в МКИ |
|  | PART\_TITLE | Название раздела |
| PIB\_CLASS | CLASS\_NNN | Уникальный идентификатор |
|  | CLASS\_INDEX | Индекс класса в МКИ |
|  | CLASS\_TITLE | Название класса |
| PIB\_SUBCLASS | SUBCLASS\_NNN | Уникальный идентификатор |
|  | SUBCLASS\_INDEX | Индекс подкласса в МКИ |
|  | SUBCLASS\_TITLE | Название подкласса |
| PIB\_GROUP | GROUP\_NNN | Уникальный идентификатор |
|  | GROUP\_INDEX | Индекс группы в МКИ |
|  | GROUP\_TITLE | Название группы |
| PIB\_SUB-GROUP | SUB-GROUP\_NNN | Уникальный идентификатор |
|  | SUB-GROUP\_INDEX | Индекс подгруппы в МКИ |
|  | SUB-GROUP\_TITLE | Название подгруппы |
| PIB\_PATENT | PATENT\_NNN | Уникальный идентификатор |
|  | PATENT\_INDEX | Патентный индекс в МКИ |
|  | PATENT\_TITLE | Название патента |
|  | PATENT\_AUTHOR | Авторы патента |
|  | PATENT\_NOTES | Примечания |
| PIB\_UDK | UDK\_NNN | Уникальный идентификатор |
|  | UDK\_INDEX | Патентный индекс в УДК |
|  | UDK\_NOTES | Примечания |
| PIB\_GRATHDOC | GRATHDOC\_NNN | Уникальный идентификатор |
|  | GRATHDOC\_FILE | Имя файла |
| PIB\_MKI | MKI\_NNN | Уникальный идентификатор |

**Исследование архитектур программно-технологической реализации АИС**

В настоящее время существует множество архитектур, служащих для разработки информационных систем, ядром которых является СУБД. Клиент в типичной конфигурации клиент/сервер - это автоматизированное рабочее место, использующее графический интерфейс (Graphical User Interface - GUI), обычно Microsoft Windows, Macintosh.

Сервер же, в основном, предназначен для хранения, передачи и распределения информации между клиентами. В клиент/серверной конфигурации программные средства имеют разделение на клиентскую и серверную часть, однако, частые обращения клиента к серверу снижают производительность работы сети и обуславливают сложность настройки системы.

Рассмотрим варианты распределения функций СУБД в клиент/серверной системы. СУБД выполняет три основные функции:

1. доступ к данным;
2. предоставление данных;
3. бизнес - функции.

Сервер СУБД может быть реализован на различных платформах, под управлением операционных систем UNIX, NetWare, Windows NT, OS/2 и др.

До появления технологии клиент/сервер большинство приложений функционировало на одной ЭВМ. Одна система отвечала не только за всю обработку данных, но также и за выполнение логики приложения. Кроме того, та же система обрабатывала весь обмен с каждым терминалом; все нажатия клавиш и элементы отображения обслуживались тем же процессором, который обрабатывал запросы к базе данных и логику приложения.

Oracle предоставляет такие возможности, как хранимые процедуры, поддержка ограничений целостности, функции, определяемые пользователем, триггеры базы данных и ряд других. Все это позволяет приложению хранить большое количество бизнес-правил (или семантику модели данных) на уровне базы данных. В результате приложение освобождается для выполнения более тонких задач обработки. Как показано на рис.28, такая СУБД намного более устойчива.

Программные продукты Oracle охватывают все основные компоненты архитектуры клиент/сервер, показанной на рис. 29:

1)полнофункциональный высокопроизводительный сервер RDBMS (система управления реляционной базой данных), масштабируемый от портативных ЭВМ до мэйнфреймов;

1. средства для разработки и запуска клиентских приложений, поддерживающие несколько сред GUI;
2. программный компонент для организации связи между БД на различных ЭВМ, который обеспечивает эффективную и безопасную связь с помощью широкого набора сетевых протоколов.

Oracle использует память системы (как реальную, так и виртуальную) для выполнения пользовательских процессов и самого программного обеспечения СУБД, и для кэширования объектов данных. В простой конфигурации Oracle файлы базы данных, структуры памяти, фоновые и пользовательские процессы располагаются на одной машине без использования сети. Однако, намного чаще встречается конфигурация, когда БД расположена на машине-сервере, а инструментальные средства Oracle - на другой машине (например, РС с Microsoft Windows). При такой клиент/серверной конфигурации машины связываются посредством некоторого сетевого программного обеспечения, которое позволяет двум машинам поддерживать связь. Для организации взаимодействия клиент/сервер или сервер-сервер необходимо использовать программный продукт Oracle SQL\*Net, который позволяет СУБД Oracle взаимодействовать с сетевым протоколом. SQL\*Net и поддерживает большинство сетевых протоколов для локальных вычислительных сетей (таких как TCP/IP, IPX/SPX) и для мэйнфреймов (например, SNA). По существу, SQL\*Net является промежуточной программной прослойкой между Oracle и сетевым ПО, обеспечивающей связь между клиентской машиной Oracle (на которой работает, например, SQL\*Plus) и сервером базы данных или между серверами баз данных. Опции SQL\*Net позволяют одной машине работать с одним сетевым протоколом, сообщаясь с другой машиной, работающей с другим протоколом.

В зависимости от размеров, таблицы (и других объектов) всех учетных разделов пользователей могут, очевидно, размещаться в одном файле базы данных, но это - не лучшее решение, так как оно не способствует гибкости структуры базы данных для управления доступом к различным пользовательским разделам, размещения базы данных на различных дисководах или резервного копирования и восстановления части базы данных. В СУБД Oracle предусмотрены привилегии системного уровня, резервное копирование и поддержка национальных языков. Все это позволяет сделать вывод о целесообразности разработки интерактивной информационной системы патентного обеспечения технологического проектирования на основе СУБД Oracle.

#### 2.3.2 Компоненты системы управления реляционной базой данных (RDBMS).

#### 2.3.2.1 Ядро системы управления реляционной базой данных (RDBMS).

Две важные части архитектуры RDBMS - *ядро*, которое является программным обеспечением, и *словарь* данных, который состоит из структур данных системного уровня, используемых ядром, управляющим базой данных.

RDBMS можно рассматривать как операционную систему (или подсистему), разработанную специально для управления доступом к данным; ее *основные функции* - хранение, выборка и обеспечение безопасности данных. Подобно операционной системе, СУБД Oracle управляет доступом одновременно работающих пользователей базы данных к некоторому набору ресурсов. *Подсистемы RDBMS* очень схожи с соответствующими подсистемами ОС и сильно интегрированы с предоставляемыми базовой ОС сервисными функциями доступа на машинном уровне к таким ресурсам, как память, центральный процессор, устройства и файловые структуры.

RDBMS поддерживают собственный список авторизованных пользователей и их привилегий; управляют кэшем памяти и страничным обменом; управляют блокировкой разделяемых ресурсов; принимают и планируют выполнение запросов пользователя; управляют использованием табличного пространства.

На рис.31. показаны основные подсистемы ядра Oracle, управляющего базой данных.

Итак, *база данных* - собрание данных, между которыми существуют (смысловые) связи. Физическое расположение и реализация базы данных прозрачны для прикладных программ; физическую базу данных можно перемещать и реорганизовывать и это не окажет влияния на работоспособность программ.

*Физически база данных Oracle* - не более чем набор файлов где-то на диске. Расположение этих файлов несущественно для функционирования (хотя важно для производительности) базы данных.

*Логически база данных* - это множество пользовательских разделов Oracle, каждый из которых идентифицируется именем пользователя (с паролем), уникальным в данной БД. На рис.29. показана архитектура Oracle.

*Существуют три основные группы файлов на диске*, составляющие базу данных.

1. Файлы базы данных
2. Управляющие файлы
3. Журнальные файлы

Наиболее важные из них - *файлы базы данных*, где располагаются собственно данные. *Управляющие и журнальные файлы* поддерживают функционирование архитектуры. Для доступа к данным БД все три набора файлов должны присутствовать, быть открытыми и доступными Oracle. Если эти файлы отсутствуют, обратиться к базе данных нельзя, и администратор базы данных должен будет восстанавливать часть или всю БД, используя файлы резервных копий (если их сделали!). Все эти файлы двоичные.

После инсталляции СУБД (об этапах установки подробно написано в [ ]) администратор имеет возможность войти в СУБД используя учетные записи SYS или SYSTEM, с парролем: master или manager, для создание учетных записей других пользовтаелей, при этом пароли учетных записей SYS и SYSTEM необходимо сразу же изменить.

Для работы с файлами базы данных на машине должны существовать системные процессы Oracle и один (или больше) пользовательский процесс.

*Системные процессы Oracle* (их называют фоновыми) обеспечивают функционирование *пользовательских процессов* - выполняют функции, которые иначе пришлось бы выполнять пользовательским процессам непосредственно.

Дополнительно к *фоновым процессам Oracle* в простейшем случае на одно подключение к базе данных должен существовать один пользовательский процесс. Пользователь должен подключиться к базе данных прежде чем он сможет обратиться к какому-либо объекту. Если один пользователь регистрируется в Oracle, используя SQL\*Plus, другой пользователь выбирает Oracle Forms, а еще один пользователь открывает электронную таблицу Excel, значит имеется три пользовательских процесса для работы с этой базой данных - по одному для каждого подключения.

Oracle использует память системы (как реальную, так и виртуальную) для выполнения пользовательских процессов и самого программного обеспечения СУБД, и для кэширования объектов данных. Существуют две главные области памяти Oracle:

1. *разделяемая память*, которая используется всеми процессами, работающими с базой данных,
2. *локальная память* для каждого пользовательского процесса.

**Системная память.**

Системная память. Oracle для всей базы данных называется SGA (system global агеа - системная глобальная область или shared global агеа - разделяемая глобальная область). Данные и управляющие структуры в SGA являются разделяемыми, и все фоновые процессы Oracle и пользовательские процессы могут к ним обращаться.

*Память пользовательского процесса.* Для каждого подключения к базе данных Oracle выделяет PGA (process global агеа - глобальную область процесса или program global агеа - глобальную область программы) в памяти машины и, кроме того, - PGA для фоновых процессов. Эта область памяти содержит данные и управляющую информацию одного процесса и между процессами не разделяется.

#### 2.3.2.2 Типы обрабатываемых данных

Типы данных обрабатываемых СУБД Oracle представлены в таблице.

Таблица 2. Типы обрабатываемых данных.

|  |  |
| --- | --- |
| *Тип данных* | *Описание* |
| СНАR(*size*) | Символьная строка фиксированной длины, имеющая максимальную длину *size* символов. Длина по умолчанию 1, максимальная -255. |
| СНАRАСТЕR(*size*) | То же, что и CHAR. |
| DATE | Правильные даты в интервале от 1 января 4712 года до н.э. до 31 декабря 4712 года. |
| LONG | Символьные данные переменной длины до 2 Гигабайт. |
| LONG RAW | Двоичные данные переменной длины вплоть до 2 Гигабайт или 231-1. |
| MLSLABEL | Используется в Trusted ORACLE. |
| NUMBER(*p,s*) | Число, имеющее *p* значащих цифр и масштаб *s. р* может быть от 1 до 38. *s* может принимать значения от -84 до 127. |
| RAW(*size*) | Двоичные данные длиной *size* байт. Максимальное значение для *size* - 2000 байт. Параметр *те для* RAW обязателен. |
| RAW MLSLABEL | Используется в Trusted ORACLE. |
| ROWID | Значения псевдостолбца ROWID. |
| VARCHAR2(*size*) | Символьная строка переменной длины, имеющая максимальную длину *size* символов. Длина по умолчанию 1, максимальная - 2000. |
| VARCHAR(*size*) | То же что и VARCHAR2. |

Извлекать данные можно также и из псевдостолбцов (табл.3), которые похожи на столбцы таблиц, но их значения нельзя изменять при помощи операторов DML.

Таблица 3. Псевдостолбцы.

|  |  |
| --- | --- |
| Название столбца | Возвращаемое значение |
| sequence.CURRVAL | Текущее значение sequence в данном сеансе (sequence.NEXTVAL должен быть выбран). |
| sequence.NEXTVAL | Следующее значение sequence в текущем сеансе. |
| [table.]LEVEL | 1 - для корня дерева, 2 - для узлов второго уровня и так далее. Используется в операторе SELECT в иерархических запросах. |
| [table.]ROWID | Значение, которое идентифицируют строку в таблице table уникальным образом. Значения псевдостолбца ROWID имеют тип данных ROWID, а не NUMBER и не CHAR. |
| ROWNUM | Порядковый номер строки среди других строк, выбираемых запросом. ORACLE выбирает строки в произвольном порядке и приписывает значения ROWNUM, прежде чем строки будут отсортированы предложением ORDER BY. |

Требования к именам объектов базы данных

* должны иметь длину от 1 до 30 бант, за исключением имен баз данных, длина которых ограничена 8 байтами;
* не могут содержать кавычек;
* не могут совпадать с именами других объектов.

Имена, которые всегда заключены в двойные кавычки, могут нарушать, приведенные ниже правила. В противном случае, имена

* должны начинаться с букв A-Z;
* могут содержать только символы A-Z, 0-9, \_, $ и #;
* не могут дублировать зарезервированные слова SQL.

Различие между прописными и строчными буквами учитывается только в именах, заключенных о двойные кавычки.

**Операции и их приоритеты**

|  |  |  |  |
| --- | --- | --- | --- |
| Арифметические операции | Символьные операции | Логические операции | Операции сравнения |
| **+ - (один операнд)** | **| |** | **NOT** | **=** |
| **\* /** |  | **AND** | **!= ^= ~= <>** |
| **+ - (два операнда)** |  | **OR** | **> >= < <=** |
|  |  |  | **IN** |
|  |  |  | **NOT IN** |
|  |  |  | **ANY, SOME** |
|  |  |  | **ALL** |