Лекция 1. Архитектура системы баз данных. Функции администратора системы баз данных в корпоративной информационной системе. Объекты базы данных.

1.1. Архитектура системы баз данных ANSI/SPARC 1
1.1.1 Внешний уровень 1
1.1.2 Концептуальный уровень 2
1.1.3 Внутренний уровень 3
1.1.4 Детализованная архитектура системы БД 3
1.2. Группа администратора базы данных (АБД) 4
1.2.1 Обязанности администратора базы данных 4
1.2.2 Обязанности сотрудника службы безопасности 4
1.2.3 Обязанности разработчика приложений 4
1.3. Основные логические объекты базы данных в современной СУБД 4
1.4 Основные физические объекты базы данных в современной СУБД 6

1.1. Архитектура системы баз данных ANSI/SPARC

состоит из 3 уровней

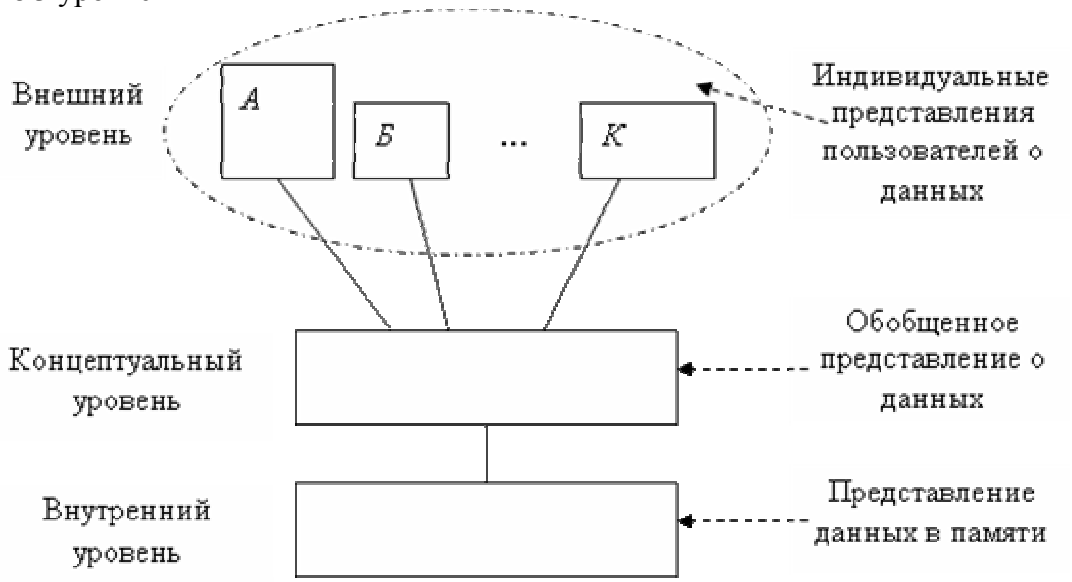

Рисунок 1 – Принципиальная архитектура системы баз данных

Как видим, в этой архитектуре учитывается, что с базой данных может работать несколько пользователей (в нашем случае – А, Б, … К, …) с разными информационными потребностями.

При этом исходят из того, что любой Банк Данных должен поддерживать разнообразные представления пользователей о Предметной Области.

> **Предметная область –** часть реального мира, представляющая интерес для данного исследования (использования).

Рассмотрим каждый уровень архитектуры подробнее.

1.1.1 Внешний уровень

Отдельного пользователя интересует, как правило, только некоторая часть всей базы данных. Представление отдельного пользователя о предметной области называется **внешним представлением**.

Таким образом, внешний уровень состоит из внешних представлений (которые в английской терминологии называются ≈views)

> **Внешнее представление** – это содержимое базы данных, каким его видит определенный пользователь.
> Состоит из множества типов **внешних записей**.

Под **записью** понимается группа взаимосвязанных элементов данных, рассматриваемых как единое целое.

Пример 1.
Пользователь из отдела кадров может рассматривать базу данных как набор записей с информацией об отделах и набор записей с информацией о служащих, и может ничего не знать о записях с информацией о деталях и поставщиках, с которыми работают пользователи из отдела поставок и сбыта.

Для использования компьютера при обработке информации о предметной области эту информацию нужно представлять в специальном виде, строго, формализовано. Формализация - неотъемлемая часть разработки любой программной системы.

Способ формального описания баз данных заключается в использовании **схем**.

> **Схема** - описание структуры БД в формализованном виде.

Схемы используются для строгого, формального описания каждого уровня архитектуры.

На внешнем уровне каждое представление пользователя описывается с помощью **внешней схемы**. Для внешних схем в общем случае используется собственный язык.

1.1.2 Концептуальный уровень

Концептуальное представление формируется на основе интеграции внешних представлений пользователей.

> **Концептуальное представление**—представление **всего** содержимого БД.

Как правило, концептуальное представление существенно отличается от внешних представлений отдельных пользователей (поскольку суммирует их разрозненные представления в одно обобщенное), и состоит из множества типов **концептуальных записей**.
Концептуальное представление определяется с помощью **концептуальной схемы**.

> **Концептуальная схема – описание полной общей логической структуры базы данных**

Концептуальная схема использует (в общем случае) другой язык описания данных. Определения концептуального языка должны относиться **только** к содержанию данных, не касаясь физических подробностей их хранения.

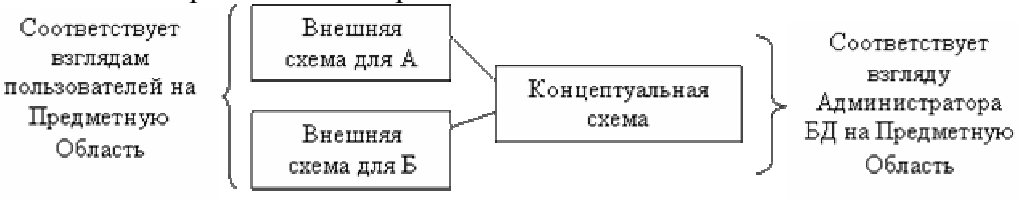

Рисунок 2 – Соответствие между взглядами пользователей и взглядом администратора БД

В концептуальной схеме **не рассматриваются** способы организации хранения или методы доступа к хранимым данным.
Определения в концептуальной схеме, помимо описания типов записей, могут включать такие средства, как безопасность, правила поддержания целостности.
Записи концептуального уровня не обязаны совпадать с записями внешних уровней.

1.1.3 Внутренний уровень

Внутреннее представление БД — представление структуры хранения записей, состоит из множества типов **хранимых записей**.

Внутреннее представление так же не связано с физическим уровнем, т.е. не рассматриваются физические записи, физические устройства хранения (например, цилиндры и дорожки цилиндры и дорожки), способы доступа к данным, расположенным удаленно.
Внутреннее представление описывается с помощью **внутренней схемы**, которая определяет не только различные типы хранимых записей, но и существующие индексы, способы представления хранимых полей, и т.д.
Внутренняя схема использует внутренний язык определения данных. Записи внутреннего уровня чаще всего не совпадают с записями внешних и концептуального уровней.

1.1.4 Детализованная архитектура системы БД

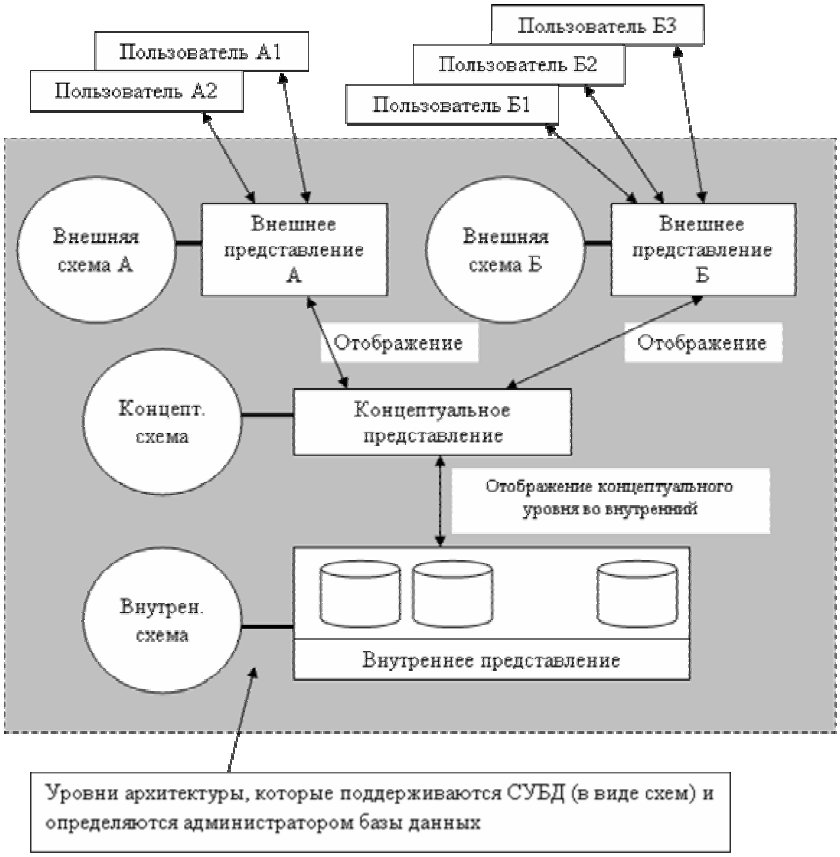

Рисунок 3 - Детализованная архитектура системы баз данных

Как видно, выбор СУБД определяет способ представления внешних, концептуальной и внутренней схем. Ответственность за корректное представление этих схем, а также за управление данными и схемами несет специальная категория пользователей – администраторы баз данных.

1.2. Группа администратора базы данных (АБД)

Поскольку система баз данных может быть весьма большой и может иметь много пользователей, должно существовать лицо или группа лиц, управляющих этой системой. Такое лицо называется **администратором базы данных** (АБД).
В любой базе данных должен быть хотя бы один человек, выполняющий административные обязанности; если база данных большая, эти обязанности могут быть распределены между несколькими администраторами.

1.2.1 Обязанности администратора базы данных

В обязанности администратора могут входить:

- инсталляция и обновление версий СУБД и прикладных инструментов
- распределение дисковой памяти и планирование будущих требований системы к памяти
- создание первичных структур памяти в базе данных (табличных пространств) по мере проектирования приложений разработчиками приложений
- создание первичных объектов (таблиц, представлений, индексов) по мере проектирования приложений разработчиками
- модификация структуры базы данных в соответствии с потребностями приложений
- зачисление пользователей и поддержание защиты системы
- соблюдение лицензионного соглашения по СУБД
- управление и отслеживание доступа пользователей к базе данных
- отслеживание и оптимизация производительности базы данных
- планирование резервного копирования и восстановления
- поддержание архивных данных на устройствах хранения информации
- осуществление резервного копирования и восстановления
- обращение за техническим сопровождением к разработчикам СУБД

1.2.2 Обязанности сотрудника службы безопасности

В некоторых случаях база данных должна также иметь одного или нескольких сотрудников службы безопасности. Сотрудник службы безопасности главным образом отвечает за регистрацию новых пользователей, управление и отслеживание доступа пользователей к базе данных, и защиту базы данных.

1.2.3 Обязанности разработчика приложений

В обязанности разработчика приложений входит:

- проектирование и разработка приложений базы данных
- проектирование структуры базы данных в соответствии с требованиями приложений
- оценка требований памяти для приложения
- формулирование модификаций структуры базы данных для приложения
- передача вышеупомянутой информации администратору базы данных
- настройка приложения в процессе его разработки
- установка мер по защите приложения в процессе его разработки

1.3. Основные логические объекты базы данных в современной СУБД.

Несмотря на достаточно простую архитектуру СБД в целом, на практике СУБД поддерживает более детальную классификацию объектов СБД, чем внешние, концептуальная и внутренняя схемы.
СУБД считает объектами СБД все доступные пользователю непосредственно компоненты структуры данных. Набор объектов обычно отличается от СУБД к СУБД, и даже от версии к версии одной СУБД. Чаще всего различают два набора объектов. Один набор объектов ведет начало от СУБД ORACLE, второй – предложен Microsoft.

Эти наборы объектов отличаются способом введения значений, за приращением которых следит СУБД – т.н. *счетчики*, или *последовательности*.

Рассмотрим набор объектов СБД, который поддерживается MS SQL Server 2000.

Таблица (table) – единственный объект СБД, в котором реально хранятся данные; являются двумерными матрицами.

Хранимая процедура (stored procedure) – набор команд SQL и Transact-SQL, сохраненный под определенным именем. Пользователи работают с таким набором, как с единым целым, обращаясь к нему по имени.

Триггер (trigger) – специальный вид хранимых процедур, который автоматически запускается сервером при удалении, изменении или добавлении записи в таблицу.

Представление (view) – виртуальная таблица, отражает результат запроса на выборку. Пользователь может работать с представлением как с обычной таблицей. С помощью представлений обычно описывается *внешняя схема* для некоторой группы пользователей.

Индекс (index) – файл специального вида, предназначен для повышения скорости работы с данными (например, для ускорения поиска данных) за счет представления этих данных в упорядоченном виде

Пользовательский тип данных (user-defined datatype) – тип данных, созданный пользователем на основе встроенных типов данных СУБД

Функция пользователя (user-defined function) – функция, создаваемая пользователем. Как и хранимая процедура, функция – тоже набор команд. В отличие от хранимой процедуры, функция может быть использована в выражениях (как, например, функция вычисления квадратного корня из значения поля).

Ограничение целостности (constraint) – объект базы данных, контролирует логическую целостность данных

Умолчание (default) – объект базы данных, который также как и ограничение целостности, может привязываться к столбцу таблицы или к пользовательскому типу данных.

Набор объектов, который поддерживается ORACLE, отличается от набора MS SQL Server в следующем:
Появляется объект «последовательность»:

Последовательность (sequence) - генерирует уникальные порядковые номера, которые могут использоваться как значения числовых столбцов таблиц базы данных.

Последовательности упрощают прикладное программирование, автоматически генерируя уникальные числовые значения для строк одной или нескольких таблиц. Номера, генерируемые последовательностью, независимы от таблиц, так что одну и ту же последовательность можно использовать для нескольких таблиц. После ее создания, к последовательности могут обращаться различные пользователи, чтобы получать действительные порядковые номера.

Под названием «программная единица» объединяют процедуры, функции и пакеты.

Процедура (procedure) – набор команд SQL и PL/SQL (языка программирования для ORACLE), который исполняется как единое целое и не возвращает результатов.

Функция (function) – набор команд SQL и PL/SQL (языка программирования для ORACLE), который исполняется как единое целое и возвращает результаты.

Пакет (package) – набор процедур и функций, сохраненный под некоторым именем.

Синоним (synonym) – это алиас (дополнительное имя) для таблицы, представлений, последовательности или программной единицы. Синоним не есть объект, но он является прямой ссылкой на объект.

Синонимы используются для:

1. маскировки действительного имени и владельца объекта
2. обеспечения общего доступа к объекту
3. достижения прозрачности местоположения для таблиц, представлений или программных единиц удаленной базы данных
4. упрощения кодирования предложений SQL для пользователей базы данных

Синоним может быть общим или личным. Индивидуальный пользователь может создать **личный синоним**, который доступен только этому пользователю. Администраторы баз данных чаще всего создают **общие синонимы**, благодаря которым объекты базовых схем становятся доступными для общего пользования всем пользователям базы данных.

Кластер (cluster) – это группа из одной или нескольких таблиц, физически хранящихся вместе. В кластеризованных таблицах связанные данные хранятся вместе, более эффективно. В некластеризованных таблицах связанные данные хранятся отдельно, занимая больше места.

Связь баз данных (database link) – именованный объект, который описывает "путь" от одной базы данных к другой. Связи баз данных неявно используются при обращении к **глобальному имени объекта** в распределенной базе данных.

Для того, чтобы управлять отдельными объектами базы данных, СУБД группирует всю информацию об этих объектах в **схеме** (schema). Схема содержит детальное описание свойств всех созданных в БД объектов.

1.4 Основные физические объекты базы данных в современной СУБД.

Все объекты и данные одной базы данных хранятся в одном или нескольких файлах данных. При планировании базы данных планируется и размещение данных по файлам. Каждая база данных состоит минимум из *двух* файлов – **файла данных** и **файла для хранения журнала транзакций** (действий с базой данных).
Каждый **файл данных** рассматривается СУБД как **набор страниц**. **Страница** – это блок фиксированного размера, который считывается с диска за одну операцию чтения. Размер страницы составляет несколько килобайт, например, в MS SQL Server 2000 размер страницы составляет 8 Кб.
Различают страницы типов **data** (только для хранения данных), **index** (для данных индекса), **text/image** (для хранения данных типа текста, изображения), **Global Allocation Map** (GAM, для хранения информации об использовании групп страниц – какому файлу какая группа страниц принадлежит), **Page Free Space** (для свободных страниц), **Index Allocation Map** (IAM, для хранения информации о самих группах страниц).
Для более эффективного управления страницами СУБД обычно объединяет группу страниц в **экстент** (extent). Размер экстента, например, в MS SQL Server 2000 – 8 страниц.