***Вопрос 2***

***Система управления базами данных*** (СУБД) - это комплекс программных и языковых средств, необходимых для создания баз данных, их поддержания в актуальном состоянии и организации в них поиска необходимой информации.

**Функции, структура и основные характеристики СУБД**

К основным функциям СУБД принято относить следующие:

* управление данными во внешней памяти;
* управление буферами оперативной памяти;
* управление транзакциями;
* журнализация и восстановление БД после сбоев;
* поддержка языков БД.

***Управление данными во внешней памяти*** включает обеспечение необходимых структур внешней памяти как для хранения данных, непосредственно входящих в базу данных, так и для служебных целей, например, для ускорения доступа к данным.

***Управление буферами оперативной памяти.*** СУБД, как правило, работают с БД большого объема. По крайней мере, объем базы данных существенно превышает объем оперативной памяти. Так что, если при обращении к любому элементу данных будет производиться обмен с внешней памятью, то вся система будет работать со скоростью устройства внешней памяти. Практически единственным способом реального увеличения этой скорости является буферизация данных в оперативной памяти. При этом, даже если операционная система производит общесистемную буферизацию, этого недостаточно для целей СУБД, которая располагает большей информацией о полезности буферизации той или иной части БД. Поэтому в развитых СУБД поддерживается собственный набор буферов оперативной памяти с собственной дисциплиной их замены.

***Управление транзакциями.*** Транзакция - это последовательность операций над БД, рассматриваемых СУБД как единое целое. Транзакция либо успешно выполняется, и СУБД фиксирует произведенные изменения данных во внешней памяти, либо ни одно из этих изменений никак не отражается на состоянии БД. Понятие транзакции необходимо для поддержания логической целостности БД, поэтому поддержание механизма транзакций является обязательным условием как однопользовательских, так и многопользовательских СУБД.

***Журнализация и восстановление БД после сбоя.*** Одним из основных требований к СУБД является надежность хранения данных во внешней памяти. Под надежностью хранения понимается то, что СУБД должна быть в состоянии восстановить последнее целостное состояние БД после любого аппаратного или программного сбоя. Обычно рассматриваются два возможных вида аппаратных сбоев: так называемые ***мягкие сбои***, которые можно трактовать как внезапную остановку работы компьютера, например, аварийное выключение питания, и ***жесткие сбои***, характеризуемые потерей информации на носителях внешней памяти. В любом из описанных случаев для восстановления БД нужно располагать некоторой избыточной информацией. Наиболее распространенным методом формирования и поддержания избыточной информации является ведение журнала изменений БД.

Журнал - это специальная служебная часть БД, недоступная пользователям, в которую поступают записи обо всех изменениях основной части БД. В виду особой важности этой информации для восстановления целостности базы данных после сбоев, важно обеспечить сверхнадежное её хранение. В некоторых СУБД поддерживаются две копии журнала, располагаемые на разных физических дисках. В разных СУБД изменения БД фиксируются на разных уровнях: иногда запись в журнале соответствует некоторой логической операции изменения БД, например, удаление строки из таблицы реляционной БД, иногда - минимальной внутренней операции модификации страницы внешней памяти, а иногда одновременно используются оба подхода. Во всех случаях придерживаются стратегии упреждающей записи в журнал. То есть, запись об изменении любого объекта БД должна попасть во внешнюю память журнала раньше, чем измененный объект попадет во внешнюю память основной части БД. Если в СУБД корректно соблюдается это условие, то с помощью журнала можно решить все проблемы восстановления БД после любого сбоя.

При мягком сбое во внешней памяти основной части БД могут находиться объекты, модифицированные транзакциями, не закончившимися к моменту сбоя, и могут отсутствовать объекты, модифицированные транзакциями, которые к моменту сбоя успешно завершились из-за использования буферов оперативной памяти, содержимое которых в этой ситуации пропадает. При соблюдении стратегии упреждающей записи, во внешней памяти журнала должна находиться информация, относящаяся к операциям модификации обоих видов объектов. Целью процесса восстановления после мягкого сбоя является состояние внешней памяти основной части БД, которое возникло бы при фиксации во внешней памяти изменений всех завершившихся транзакций и которое не содержало бы никаких следов незаконченных транзакций. Для того чтобы этого добиться, сначала производят откат незавершенных транзакций, а потом повторно воспроизводят те операции завершенных транзакций, результаты которых не отображены во внешней памяти.

Для восстановления БД после жесткого сбоя используют журнал и архивную копию БД. Архивная копия является полной копией БД к моменту начала заполнения журнала. Восстановление БД состоит в том, что, исходя из архивной копии, по журналу воспроизводится работа всех транзакций, которые закончились к моменту сбоя.

***Поддержка языков БД.*** Для работы с базами данных используются специальные языки, в целом называемые языками баз данных. В ранних СУБД поддерживалось несколько специализированных по своим функциям языков. Чаще всего выделялись два языка - язык определения схемы БД (SDL - Schema Definition Language) и язык манипулирования данными (DML - Data Manipulation Language). SDL служил, главным образом, для определения логической структуры БД, какой она представляется пользователям. DML содержал набор операторов манипулирования данными, позволяющих вводить, удалять, модифицировать и выбирать данные. В современных СУБД, обычно, поддерживается единый интегрированный язык, содержащий все необходимые средства для работы с БД и обеспечивающий базовый пользовательский интерфейс. Стандартным языком наиболее распространенных в настоящее время реляционных СУБД является язык SQL (Structured Query Language).

Язык SQL содержит специальные средства определения ограничений целостности БД. Ограничения целостности хранятся в специальных таблицах-каталогах. Обеспечение контроля целостности производится на языковом уровне. При компиляции операторов модификации БД, компилятор SQL, на основании имеющихся ограничений целостности, генерирует соответствующий программный код.

Специальные операторы языка SQL позволяют определять так называемые представления БД, фактически являющиеся хранимыми запросами. Для пользователя представление является такой же таблицей, как любая базовая таблица, хранимая в БД, но с его помощью можно ограничить или расширить видимость БД для конкретного пользователя. Поддержание представлений производится также на языковом уровне.

Наконец, авторизация доступа к объектам БД производится на основе специального набора операторов SQL. Идея состоит в том, что для выполнения операторов SQL разного вида пользователь должен обладать различными полномочиями. Пользователь, создавший таблицу БД, обладает полным набором полномочий для работы с этой таблицей. В число таких полномочий входит право на передачу всех или части полномочий другим пользователям, включая полномочие на передачу полномочий. Полномочия пользователей описываются в специальных таблицах-каталогах, а контроль полномочий поддерживается на языковом уровне.

В типовой структуре современной реляционной СУБД логически можно выделить ядро СУБД, компилятор языка БД, подсистему поддержки времени выполнения и набор утилит.

***Ядро СУБД*** отвечает за управление данными во внешней памяти, управление буферами оперативной памяти, управление транзакциями и журнализацию. Соответственно, можно выделить такие компоненты ядра как менеджер данных, менеджер буферов, менеджер транзакций и менеджер журнала. Ядро обладает собственным интерфейсом, недоступным пользователям, и является основной резидентной частью СУБД. При использовании архитектуры «клиент-сервер» ядро является основной составляющей серверной части системы.

Основной функцией ***компилятора языка БД*** является преобразование операторов языка БД в выполняемую программу. Основной проблемой реляционных СУБД является то, что языки этих систем являются непроцедурными, то есть в операторе такого языка специфицируется некоторое действие над БД, но эта спецификация не является процедурой, а лишь описывает в некоторой форме условия совершения желаемого действия. Поэтому компилятор должен решить, каким образом выполнять оператор языка прежде, чем произвести программу. Применяются достаточно сложные методы оптимизации операторов. Выполняемая программа представляется в машинных кодах или в выполняемом внутреннем машинно-независимом коде. В последнем случае реальное выполнение оператора производится с привлечением ***подсистемы поддержки времени выполнения***, представляющей собой интерпретатор этого внутреннего языка.

В отдельные ***утилиты*** обычно выделяют такие процедуры, которые слишком сложно выполнять с использованием языка БД, например, загрузка и выгрузка БД, сбор статистики, глобальная проверка целостности и другие. Утилиты программируются с использованием интерфейса ядра СУБД.

***Производительность СУБД оценивается:***

* временем выполнения запросов;
* скоростью поиска информации в неиндексированных полях;
* временем выполнения операций импортирования базы данных из других форматов;
* скоростью создания индексов и выполнения таких массовых операций, как обновление, вставка, удаление данных;
* максимальным числом параллельных обращений к данным в многопользовательском режиме;
* временем генерации отчета.

**Типы моделей данных**

Основой любой базы данных является реализованная в ней модель данных, представляющая собой множество структур данных, ограничений целостности и операций манипулирования данными. С помощью модели данных могут быть представлены объекты предметной области и существующие между ними связи.

Базовыми моделями представления данных являются ***иерархическая***, ***сетевая*** и ***реляционная***.

***Иерархическая модель*** данных представляет информационные отображения объектов реального мира – сущности и их связи в виде ориентированного графа или дерева. К основным понятиям иерархической структуры относятся уровень, элемент или узел и связь. Узел - это совокупность атрибутов, описывающих некоторый объект. На схеме иерархического дерева узлы представляются вершинами графа. Каждый узел на более низком уровне связан только с одним узлом, находящимся на более высоком уровне. Иерархическое дерево имеет только одну вершину (корень дерева), не подчиненную никакой другой вершине и находящуюся на самом верхнем (первом) уровне. Зависимые (подчиненные) узлы находятся на втором, третьем и так далее уровнях. Количество деревьев в базе данных определяется числом корневых записей.

К каждой записи базы данных существует только один (иерархический) путь от корневой записи.

Примерами операторов манипулирования иерархически организованными данными могут быть следующие:

* найти указанное дерево БД;
* перейти от одного дерева к другому;
* перейти от одной записи к другой внутри дерева;
* перейти от одной записи к другой в порядке обхода иерархии;
* вставить новую запись в указанную позицию;
* удалить текущую запись.

В иерархической модели данных автоматически поддерживается целостность ссылок между предками и потомками. Основное правило: никакой потомок не может существовать без своего родителя.

***Сетевая модель*** организации данных является расширением иерархической модели. В иерархических структурах запись-потомок должна иметь только одного предка - в сетевой структуре данных потомок может иметь любое число предков.

Понятие ***реляционной модели данных*** (от английского *relation* - отношение) связано с разработками Е. Кодда. Эти модели характеризуются простотой структуры данных, удобным для пользователя табличным представлением и возможностью использования формального аппарата реляционной алгебры и реляционного исчисления для обработки данных.

Реляционная модель ориентирована на организацию данных в виде двумерных таблиц. Реляционная таблица представляет собой двумерный массив и обладает следующими свойствами:

* каждый элемент таблицы — один элемент данных;
* все столбцы в таблице однородные, то есть, все элементы в столбце имеют одинаковый тип (числовой, символьный или другой) и длину;
* каждый столбец имеет уникальное имя;
* одинаковые строки в таблице отсутствуют;
* порядок следования строк и столбцов может быть произвольным.

Отношения представлены в виде таблиц, строки которых соответствуют кортежам или записям а столбцы - атрибутам отношений, доменам, полям.

Поле, каждое значение которого однозначно определяет соответствующую запись, называется простым ключом. Если записи однозначно определяются значениями нескольких полей, то такая таблица базы данных имеет составной ключ.

Чтобы связать две реляционные таблицы, необходимо ключ первой таблицы ввести в состав ключа второй таблицы или ввести в структуру первой таблицы внешний ключ - ключ второй таблицы.

В реляционной модели данных фиксируются два базовых требования целостности, которые должны поддерживаться в любой реляционной СУБД. Первое требование называется требованием целостности сущностей, которое состоит в том, что любой кортеж любого отношения должен быть отличим от любого другого кортежа этого отношения, то есть любое отношение должно содержать первичный ключ.

Второе требование называется требованием целостности по ссылкам и состоит в том, что для каждого значения внешнего ключа в отношении, на которое ведет ссылка, должен найтись кортеж с таким же значением первичного ключа, либо значение внешнего ключа должно быть неопределенным.

В качестве операторов манипулирования данными в реляционных моделях используются операторы языка структурированных запросов SQL.