**1) Информационная система**— это взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели.

* информационные системы оперативного (операционного) уровня – бухгалтерская, банковских депозитов, обработки заказов, регистрации билетов, выплаты зарплаты;
* информационная система специалистов – офисная автоматизация, обработка знаний (включая экспертные системы);
* информационные системы тактического уровня (среднее звено) – мониторинг, администрирование, контроль, принятие решений;
* стратегические информационные системы – формулирование целей, стратегическое планирование.

**7) Классификация программ СУБД.**

* 1) полнофункциональные СУБД; Обычно ПФСУБД имеют развитый интерфейс, позволяющий с помощью команд меню выполнять основные действия с БД: создавать и модифицировать структуры таблиц, вводить данные, формировать запросы, разрабатывать отчеты, выводить их на печать и т. п. Microsoft Access, Microsoft FoxPro.
* 2) серверы БД; Серверы БД реализуют функции управления базами данных, запрашиваемые другими (клиентскими) программами обычно с помощью операторов SQL.InterBase (Borland).
* 3) клиенты БД; для серверов БД в общем случае могут использоваться различные программы: ПФСУБД, электронные таблицы, текстовые процессоры, программы электронной почты и т. д.
* 4) средства разработки программ работы с БД. К средствам разработки пользовательских приложений относятся системы программирования, разнообразные библиотеки программ для различных языков программирования, а также пакеты автоматизации разработок (в том числе систем типа клиент-сервер). Visual Basic (Microsoft)

**9) Реляционная алгебра — замкнутая система операций над**[**отношениями**](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BD%D0%BE%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_(%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C))**в**[**реляционной модели данных**](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)**.(Объединение пресечение вычитание проекция соединение деление произведение)**

**23) SELECT \* FROM table where….;**

**24)** Оператор **IN** определяет набор значений, в которое данное значение может или не может быть включено. Where birthfday in (1999, 2000,2001,2004);

**BETWEEN** определяет диапазон, значения которого делают предикат верным.  BETWEEN 1964 AND 1975

Оператор **LIKE** необходим, если Вам потребуется поиск в столбцах таблицы, которые имеют тип CHAR или VARCHAR WHERE Имя LIKE "А%";

Вывести все поля из талицы Customers, значения поля city которых равны NULL:

SELECT \* FROM Customers WHERE city **IS** NULL

**25**) грегирующие функции позволяют получать из таблицы сводную (агреги­рованную) информацию, выполняя операции над группой строк таблицы. Для задания в SELECT-запросе агрегирующих операций используются следующие ключевые слова:

* **count** определяет количество строк или значений поля, выбранных по­средством запроса и не являющихся значениями;
* **sum** вычисляет арифметическую сумму всех выбранных значений дан­ного поля;
* **avg** вычисляет среднее значение для всех выбранных значений/данною поля;
* **мах** вычисляет наибольшее из всех выбранных значений данного поля;
* **min** вычисляет наименьшее из всех выбранных значений данного поля.

**26)** Предикат **EXISTS** принимает значение TRUE, если подзапрос содержит любое количество строк, иначе его значение равно FALSE. Найти тех производителей портативных компьютеров, которые также производят принтеры: WHERE type = 'laptop' AND EXISTS (SELECT maker FROM Product where…)

**28)** Вы можете поместить многочисленные запросы вместе и объединить их вывод используя предложение UNION. Предложение UNION объединяет вывод двух или более SQL запросов в единый набор строк и столбцов. Например чтобы получить всех продавцов и заказчиков размещенных в Лондоне и вывести их как единое целое вы могли бы ввести:

SELECT snum, sname FROM Salespeople WHERE city = 'London' UNION SELECT cnum, cname FROM Customers WHERE city = 'London';

**29)** Ну емае ну join дату сразу из нескольких таблиц тянем

SELECT dbo.Orders.id, dbo.Orders.nane, dbo.Products.name

FROM dbo.Orders LEFT OUTER JOIN

dbo.Products ON dbo.Orders.product\_id = dbo.Products.id