**1)Цель, особенности и работа с представлениями**

Представления можно использовать для разных целей:

Для ограничения использования определенных столбцов и/или строк таблиц.

Для скрытия подробностей сложных запросов.

Для ограничения вставляемых или обновляемых значений некоторым диапазоном.

обязательно требуется явно указывать имена столбцов:

если столбец представления создается из выражения или агрегатной функции

если два или больше столбцов представления имеют одинаковое имя в базовой таблице.

CREATE VIEW view\_Count(projectNumber, countProject)

AS SELECT ProjectNumber, COUNT(\*)

FROM Works\_on

GROUP BY ProjectNumber;

**2)Понятие триггеров, назначение, примеры**

alter TRIGGER insert\_Table\_1 ON Table\_1

after insert \ INSTEAD OF INSERT

AS

BEGIN

declare @rez varchar(max);

select @rez=(select department from inserted\deleted)

insert\update\delete into Log\_table values (@rez, GETDATE())

END

**3)Понятие хранимых процедур, назначение, примеры.**

create PROCEDURE count\_trip\_drivers @Id INT, @count INT OUTPUT

AS SELECT @count = COUNT(\*)

FROM trips

WHERE driver\_id = @Id

declare @xxx INT

exec count\_trip\_drivers @Id = 3, @count = @xxx OUTPUT;

print @xxx

**4)Пользовательские функции, назначение, примеры**

в функции не проходит GETDATE ( )

функция может использоваться в составе SQL-запроса

CREATE FUNCTION drivers\_with\_experiense

(@experiense int)

RETURNS TABLE

AS

RETURN

(

SELECT \*from drivers where experience >= @experiense

)

GO

select \* from drivers\_with\_experiense(10)

**5)Понятие и виды индексов, назначение, примеры**

Читатель книги имеет возможность самому решать, использовать ли индекс в каждом конкретном случае или нет. Пользователь базы данных такой возможности не имеет, и за него это решение принимает компонент системы, называемый оптимизатором запросов.

Индекс для определенной книги создается вместе с книгой, после чего он больше не изменяется. В противоположность, индекс базы данных может меняться при каждом изменении соответствующих данных.

Кластеризованный индекс определяет физический порядок данных в таблице.

Структура некластеризованного индекса точно такая же, как и кластеризованного, но с двумя важными отличиями:

некластеризованный индекс не изменяет физическое упорядочивание строк таблицы;

страницы узлов некластеризованного индекса состоят из ключей индекса и закладок.

В случае операции соединения рекомендуется создавать индекс для каждого соединяемого столбца. Соединяемые столбцы часто представляют первичный ключ одной из таблицы и соответствующий внешний ключ другой таблицы. Если указываются ограничения для обеспечения целостности PRIMARY KEY и FOREIGN KEY для соответствующих соединяемых столбцов, следует создать только некластеризованный индекс для столбца внешнего ключа, т.к. система неявно создаст кластеризованный индекс для столбца первичного ключа.

**6)Модели и подходы к защите данных в MS SQL**

Роли, имена для входа, разрешение действий.. тру ля ля тополя

Не забыть про представления и процедуры

**7)Основные виды, подходы к резервному копированию и восстановлению**

Запрещено во время резервного копирования:

Создание и изменение баз данных

Создание индексов

Разновидности резервного копирования:

Полное резервное копирование базы данных

Разностное резервное копирование

Резервное копирование журнала транзакций

Резервное копирование файла или группы файлов базы данных

Стратегии резервного копирования:

Стратегия полного резервного копирования базы данных

Стратегия полного резервного копирования базы данных с журналом транзакций

Стратегия разностного резервного копирования

Стратегия резервного копирования файлов или групп файлов базы данных

**8)Способы распространения данных**

Публикация

Собрание одной или нескольких статей

Основа подписки

В базе данных можно создать одну или несколько публикаций

Статья

Отдельная таблица или подмножество данных таблицы

Часть публикации

Подписки: принудительная и по запросу

**типы репликации**

Репликация моментальных снимков

Периодическая массивная передача новых моментальных снимков данных

Репликация транзакций

Репликация произошедших изменений

Репликация слиянием

Автономные изменения реплицированных данных впоследствии объединяются

Модели репликации:

Обзор моделей репликации

Использование комбинированных моделей и типов репликации

Пример модели с центральным издателем и удаленным распределителем

Пример модели с центральным подписчиком и несколькими издателями

Пример модели с несколькими издателями и несколькими подписчиками

**9)Сценарии репликации**

**10)Обмен данными MS SQL**

Партицирование: это разбиение больших таблиц на логические части по выбранным критериям.

По диапазону, по ключу, по значение, по хэшу

Репликация

Шардинг - развитие партиционирования - разбиение данных на группы и хранение каждой группы на отдельном сервере (шарде). В данном случае группа не обязательно включает одну таблицу, это может быть несколько таблиц содержащих одно целое.

Выртикальный и горизонтальный

**11)Утилита SQL Server Agent**