МГТУ ИМ. Н.Э. БАУМАНА

Базы Данных

Семинар 1

БД – самодокументированное собрание интегрированных записей.

Самодокументированное – Содержит описание собственной структуры -> Это словарь данных, каталог данных, метаданные.

Интегрированных – Файлы данных, метаданные, индексы

Требования к организации БД:

1. Не избыточность данных. Каждый элемент данных присутствует в БД в единственном экземпляре.
2. Совместное использование данных многими пользователями.
3. Эффективность доступа к БД. Высокое быстродействие, малое время отклика.
4. Целостность данных. Сохранение внутренней логики, структуры, соблюдение явно заданных правил.
5. Безопасность данных. Защита от преднамеренного или непреднамеренного искажения или разрушения.
6. Восстановление данных после программных и аппаратных сбоев.
7. Независимость данных от других прикладных программ.

СУБД – приложение, обеспечивающее создание, хранение, обновление и поиск информации в БД.

Основные функции СУБД:

1. Управление данными во внешней памяти
2. Управление операционной памятью – буферизация данных в ОП
3. Управление транзакциями
4. Журнализация
5. Поддержка языков БД (SQL)

Основные компоненты СУБД:

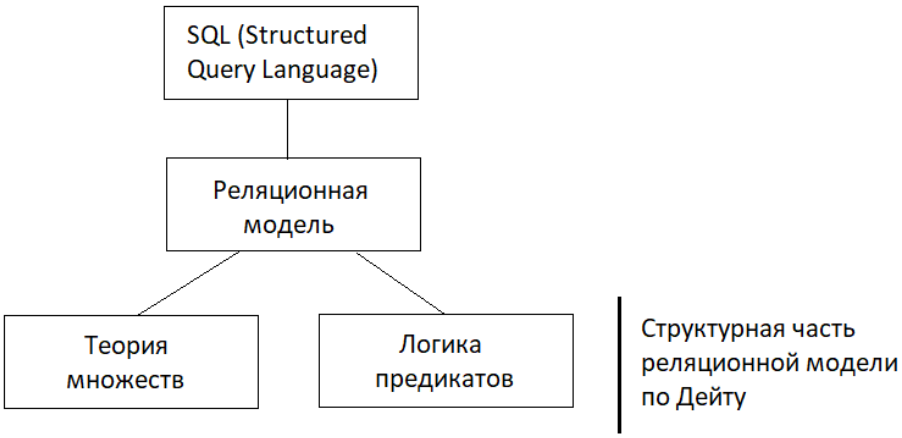
1. Ядро. Управление данными во внешней и оперативной памяти + журнализация.
2. Процессор языка БД. Оптимизация запросов на изменение и извлечение данных и создание исполняемого кода.

3. Подсистема поддержки времени использования. Интерпретирует программы манипуляции данными, создающие пользовательский интерфейс с БД.
4. Сервисные программы. Дополнительные возможности по обслуживанию ИС.

ОСНОВЫ ЯЗЫКА T-SQL

SQL – стандарт международной организации по стандартизации ISO (SQL:2011) и американского национального института стандартов (ANSI).

T-SQL – диалект стандартного языка SQL.



В основе стандарта лежит реляционная модель – математическая модель для управления и обработки данных. Реляционная модель оперирует понятием «Отношение». Отношение имеет заголовок и тело.

Заголовок – набор атрибутов (В SQL - столбцы), каждый из которых имеет определенный тип. Атрибут – совокупность имени и типа данных.

Тело – множество картежей (В SQL – строки). Заголовок кортежа – заголовок отношения.

Логику работы с данными можно разделить на три основных языка:

1. DDL (Data Definition Language). Служит для описания структуры БД:
    - Создание объекта (create)
    - Изменение объекта (alter)
    - Удаление объекта (drop)
2. DML (Data Manipulation Language). Служит для выполнения операция с данными:
    - Выбор информации (select)
    - Добавление данных (insert)
    - Обновление данных (update)
    - Удаление данных (delete)
3. DCL (Data Control Language). Служит для осуществления администраторских действий
    - Выдача прав доступа к объекту (grant)
    - Удаление прав доступа на объект (revoke)

DDL

Создание и изменение таблиц.

Временные таблицы. Существуют на протяжении сессии БД (например, пока не закрыли окно запросов). Работа с командами аналогична обычным таблицам. Название начинается в символа «#». Если используется 1 знак «#», то создается локальная таблица, доступная только в текущей сессии. Если используется 2 знака, создается глобальная временная таблица, доступная всем открытым сессиям БД.

```
CREATE TABLE #table1
```

Табличные переменные. Переменные с особым типом данных TABLE, используются для временного хранения результирующего набора данных в виде строк таблицы. Создаются с помощью инструкции DECLARE. Переменные такого типа служат альтернативой временных таблиц. Табличные переменные нельзя изменить после их создания или создать с помощью выборки из другой таблицы, также не изменяются в результате откатов транзакция, т.к. не являются частью постоянных данных БД.

```
DECLARE @Table2 (…);
```

Представление. Виртуальные таблицы, содержащие запросы, динамически извлекающие используемые данные. Изменение данных в реальной таблице БД отображаетеся в содержимом представления при первом же обращении.

```
CRAETE VIEW View1 () AS …;
```

Индексированное представление. Представления, для которых создан кластеризованный индекс. Создание такого индекса означает, что система материализует динамические данные в страницах узлов структуры индекса. Результирующий набор, возвращаемый представлением с кластеризованным индексом, сохраняется в БД таким же образом как и таблица с кластеризованным индексом.

```
CRAETE VIEW View2 () AS …;
CREATE UNIQUE CLUSTERED INDEX index_name ON View2;
```

Таблица. Основной способ хранения данных.

Схема обращения к таблице: [название БД].[название схемы].Название таблицы.

Явный способ создания таблиц.

```
CREATE TABLE [database_name.[schema_name].|schema_name. ] table_name [ AS
FileTable ](
        { <column_definition>
      |<computed_column_definition>
      |<column_set_definition>
      |[<table_constraint>]
         |[<table_index> ] [ ,...n ]
        }
)
[ON {partition_scheme_name (partition_column_name) | "default"}]
[ WITH ( <table_option> [ ,...n ] ) ]
[ ; ]
```

Атрибут – название столбца, его тип + дополнительные настроки: допускаются/не допускаются NULL значения или в это столбец-счетчик + могут присутствовать дополнительные ограничения на столбец (уникальность/первичный ключ/…).Например, MyColumn int not null

```
<column_definition> ::= column_name <data_type>
        [NULL | NOT NULL]
        |[IDENTITY [(seed, increment)] [NOT FOR REPLICATION]]
[<column_constraint> [ ...n ] ]
```

Тип – название типа (если нужно, еще и название схемы), размер, точность, максимальное значение, XML-коллекция. Например, varchar(100), decimal(15,3)

```
<data _type> ::= [type_schema_name.] type_name
 [ (precision [ ,scale]
        | max
        | [{ CONTENT | DOCUMENT }] xml_schema_collection)
 ]
```

Ограничения – правила, принудительно применяющиеся к компонентам БД (таблицы/столбцы/…). Могут иметь имя, могут содержать логическое выражение для проверки соедржмиого атрибута. Бывают кластеризованные и не кластеризованные. Например: PRIMARY KEY (contact_id, group_id), FOREIGN KEY (contact_id) REFERENCES contacts (contact_id) ON DELETE CASCADE ON UPDATE NO ACTION

```
<column_constraint> ::= [CONSTRAINT constraint_name]
  {   {PRIMARY KEY | UNIQUE} [CLUSTERED | NONCLUSTERED]
        [  WITH FILLFACTOR = fillfactor
          |WITH (< index_option > [ , ...n ])
        ]
          [ ON {partition_scheme_name (partition_column_name)
                    | filegroup
                    | "default" }
          ]
          | [ FOREIGN KEY ] REFERENCES [schema_name.]
                 referenced_table_name [(ref_column)]
          [ON DELETE {NO ACTION|CASCADE|SET NULL|SET DEFAULT}]
          [ON UPDATE {NO ACTION|CASCADE|SET NULL|SET DEFAULT}]
          [NOT FOR REPLICATION]
          |CHECK [NOT FOR REPLICATION] (logical_expression)
}
```

```
< table_constraint > ::= [CONSTRAINT constraint_name]
{
    { PRIMARY KEY | UNIQUE }
    [CLUSTERED | NONCLUSTERED]
    (column [ASC | DESC] [ ,...n ] )
    [
        WITH FILLFACTOR = fillfactor
        | WITH ( <index_option> [ , ...n ])
    ]
    [ON {partition_scheme_name (partition_column_name)
            |filegroup
            |"default"
        }
    ]
    | FOREIGN KEY (column [ ,...n ])
      REFERENCES referenced_table_name [(ref_column [ ,...n ])]
    [ON DELETE {NO ACTION|CASCADE|SET NULL|SET DEFAULT}]
    [ON UPDATE {NO ACTION|CASCADE|SET NULL|SET DEFAULT}]
    [ NOT FOR REPLICATION ]
    | CHECK [NOT FOR REPLICATION] ( logical_expression )
}
```

Свойствао вычисляемого столбца. Определяется как название атрибута и способ его получения.Вычисляемый столбец является виртуальным столбцом, который физически не хранится в таблице, а вычисляется с помощью некоторого выражения, использующего другие столбцы данной таблицы.Например: cost AS price*qty

```
<computed_column_definition> ::= column_name AS
    computed_column_expression
    [PERSISTED [NOT NULL]]
    [ [CONSTRAINT constraint_name] {PRIMARY KEY | UNIQUE}
      [CLUSTERED | NONCLUSTERED]
      [ WITH FILLFACTOR = fillfactor
        | WITH (<index_option> [ , ...n ])
      ]
      [ON { partition_scheme_name (partition_column_name)
                                  | filegroup
                                  | "default"
          }
      ]
      | [FOREIGN KEY]
      REFERENCES referenced_table_name [ (ref_column ) ]
      [ON DELETE {NO ACTION|CASCADE}]
      [ON UPDATE {NO ACTION}]
      [NOT FOR REPLICATION]
      |CHECK [ NOT FOR REPLICATION ] (logical_expression )
   ]
```

Дополнительные опции при создании таблицысновной задачей которых является оптимизация хранения и доступа к таблице.Позволяет сжимать данные, применять сжатие для конкретных блоков таблицы (партици), указывать специфисечкие имена при использовании первичных ограничений, оргничений уникальности.

```
<table_option> ::= {
   [DATA_COMPRESSION = {NONE|ROW|PAGE}
         [ON PARTITIONS ({<partition_number_expression>|
<range>}
             [ , ...n ])
         ]
     ]
     [FILETABLE_DIRECTORY = <directory_name>]
     [FILETABLE_COLLATE_FILENAME = {<collation_name>
database_default}]
     [FILETABLE_PRIMARY_KEY_CONSTRAINT_NAME =
<constraint_name>]
     [FILETABLE_STREAMID_UNIQUE_CONSTRAINT_NAME =
<constraint_name>]
     [FILETABLE_FULLPATH_UNIQUE_CONSTRAINT_NAME =
<constraint_name>]
}
```

Пример создания таблицы:

```
CREATE TABLE dbo.EmployeePhoto
(
   Id int IDENTITY(1, 1),
   EmployeeId int NOT NULL PRIMARY KEY,
   Photo varbinary(max) FILESTREAM NULL,
   MyRowGuidColumn uniqueidentifier NOT NULL ROWGUIDCOL
      UNIQUE DEFAULT NEWID()
);
```