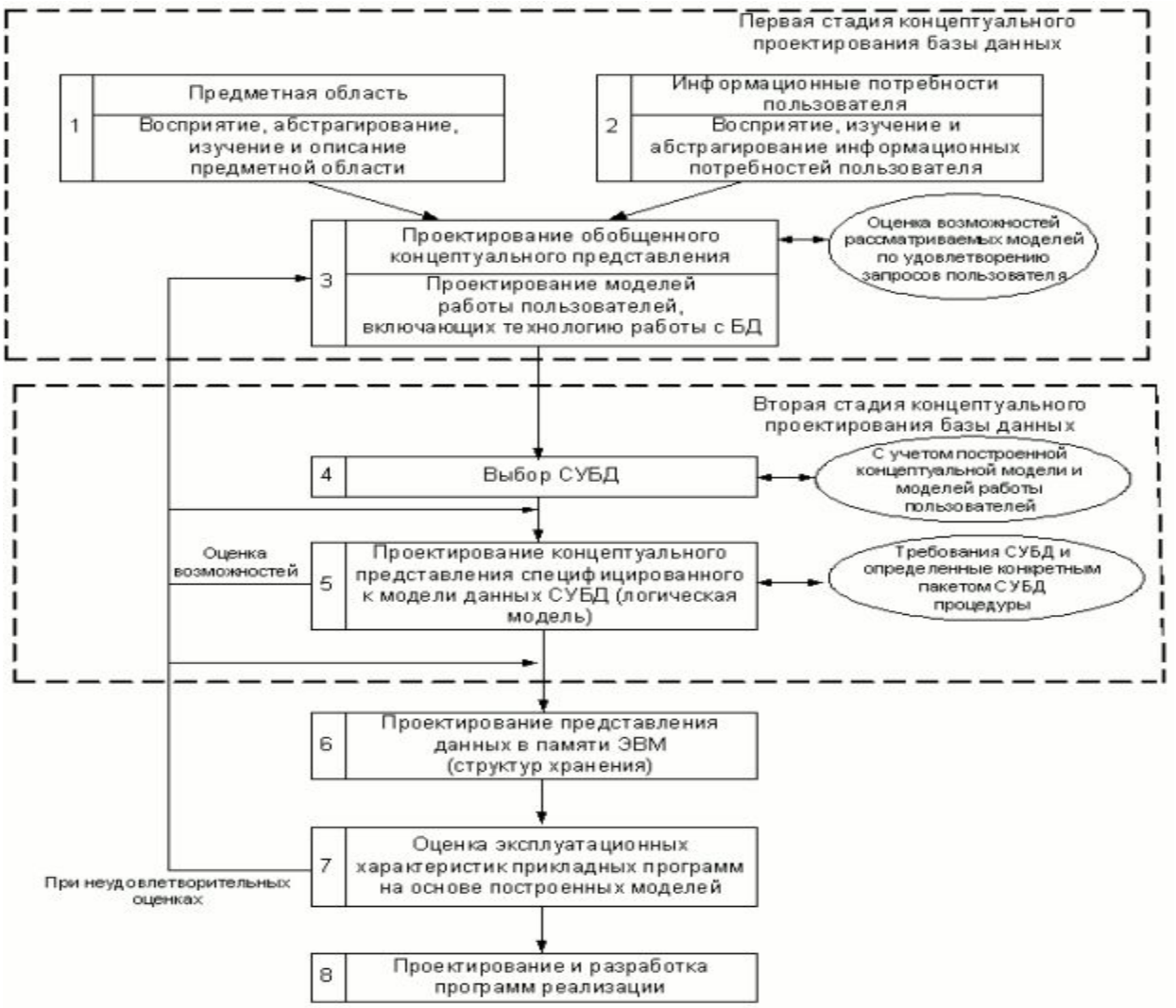
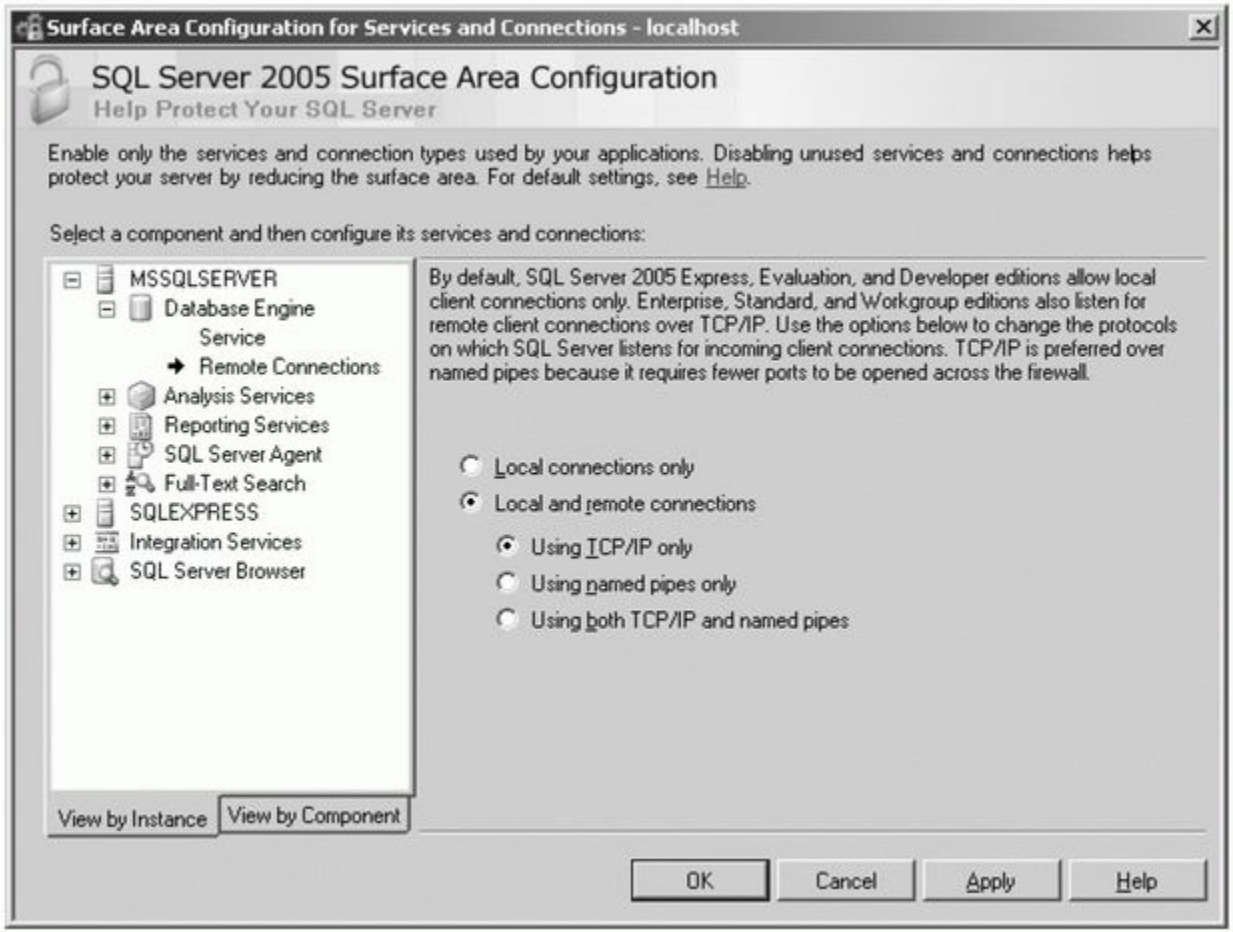
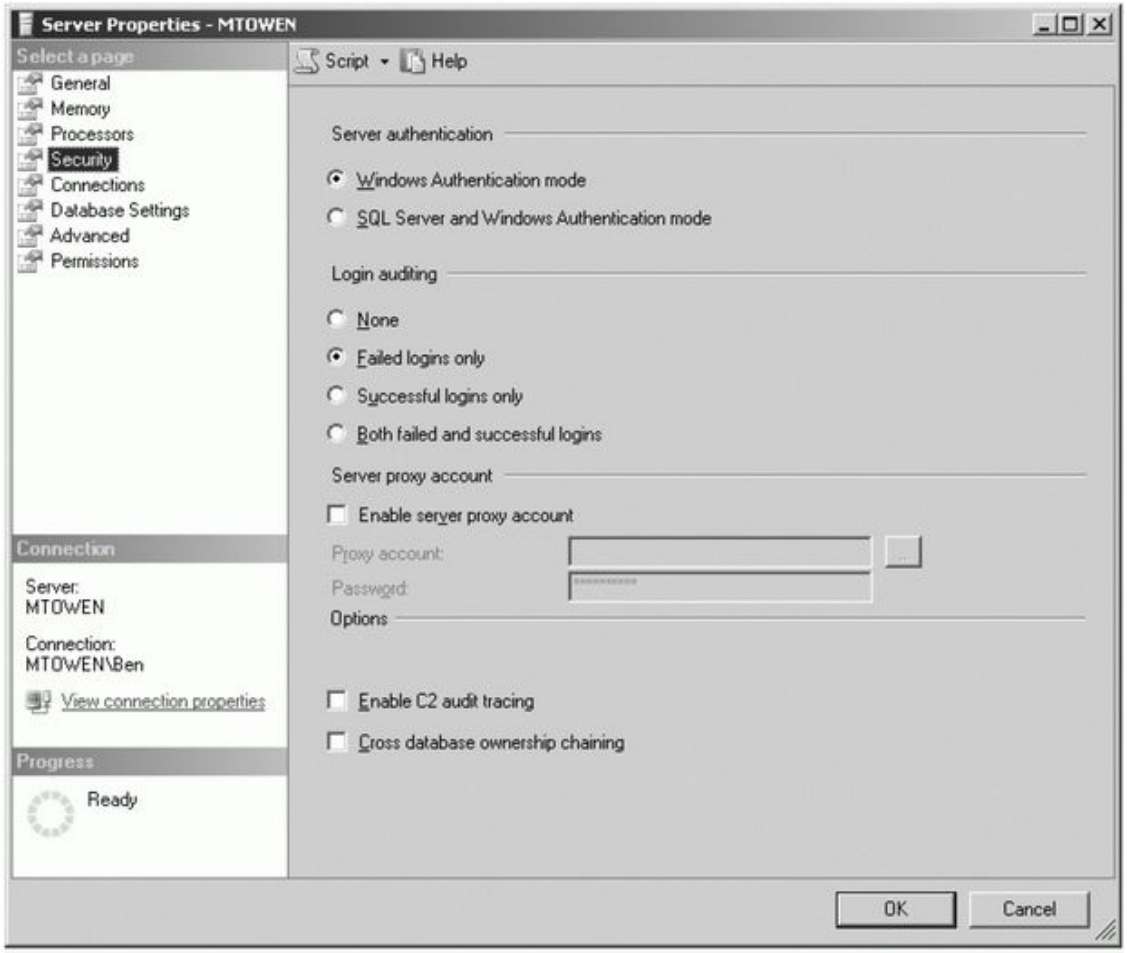
Этапы проектирования БД



Удаленные соединения с БД



Установка режима проверки подлинности



Установление паролей

- Парольную политику можно включить при помощи следующего кода Transact SQL:
- CREATE LOGIN Mary
- WITH PASSWORD = '34TY$$543'
- MUSTCHANGE,
- CHECK EXPIRATION = ON,
- CHECK POLICY = ON;

Управление доступом к базам данных

- **Добавляем пользователя базы данных**
- Добавить пользователя базы данных можно при помощи инструкции CREATE USER. Следующий пример кода Transact-SQL создает имя входа Peter и сопоставленного с ним пользователя в базе данных Adventure Works:
- Создаем имя входа Peter
- CREATE LOGIN Peter WITH PASSWORD='Tyu87IOR0';
- Изменяем контекст соединения на базу данных AdventureWorks.
- USE AdventureWorks;
- GO
- Добавляем пользователя базы данных Peter, который сопоставлен имени входа Peter в БД AdventureWorks.
- CREATE USER Peter FOR LOGIN Peter;
- **Управляем пользователями базы данных**
- Проверить, имеет ли текущее имя входа доступ к базе данных, можно при помощи следующей инструкции:
- SELECT HAS_DBACCESS('AdventureWorks');
- Чтобы получить информацию о пользователях базы данных, можно воспользоваться представлением каталога sys.database_principals.
- Если нужно временно отключить доступ пользователя к базе данных, можно отозвать разрешение CONNECT для этого пользователя. Следующий пример отзывает разрешение CONNECT для пользователя Peter:
- Изменяем контекст соединения на базу данных AdventureWorks.
- USE AdventureWorks;
- GO
- Отзываем разрешение connect для Peter в AdventureWorks.
- REVOKE CONNECT TO Peter;

Администрирование БД, управление ролями

- Выполнив запрос системной функции IS_MEMBER, можно проверить, принадлежит ли текущий пользователь к какой-либо роли базы данных. В следующем примере мы проверим, принадлежит ли текущий пользователь к роли базы данных db_owner.
- Изменяем контекст соединения на базу данных AdventureWorks.
- USE AdventureWorks;
- GO
- Проверяем, принадлежит ли текущий пользователь к роли db_owner
- SELECT IS_MEMBER ("db_owner");
- **Совет**. Системную функцию IS_MEMBER можно использовать для проверки того, принадлежит ли текущий пользователь также к определенной группе Windows, как показано в следующем примере:
  - Изменяем контекст соединения на базу данных AdventureWorks.
  - USE AdventureWorks;
- GO
  - Проверяем, принадлежит ли текущий пользователь к группе Managers в домене ADVWORKS
- SELECT IS_MEMBER ("[ADVWORKS\Managers]");
- Удалить пользователей из роли базы данных можно при помощи системной хранимой процедуры sp_droprolemember. Если нужно удалить роль базы данных, то используется инструкцияDROP ROLE. Следующий код удаляет пользователя Peter из роли базы данных Auditors, а затем удаляет роль Auditors.
- Изменяем контекст соединения на базу данных AdventureWorks.
- USE AdventureWorks;
- GO
- Удаляем пользователя Peter из роли Auditors
- EXECUTE sp_droprolemember "Auditors", "Peter";
- Удаляем роль Auditors из текущей базы данных
- DROP ROLE Auditors;

Управление доступом к схемам

- В SQL Server 2005 реализована концепция ANSI для схем. Схемы - это контейнеры объектов, которые позволяют группировать объекты базы данных. Схемы оказывают большое влияние на то, как пользователи ссылаются на объекты базы данных. В SQL Server 2005 объект базы данных называется именем, состоящим из четырех компонентов следующей структуры:
- <Server>.<Database>.<Schema>.<Object>.

- Изменяем контекст соединения на базу данных AdventureWorks.
- USE AdventureWorks;
- GO
- Создаем схему Accounting с владельцем Peter.
- CREATE SCHEMA Accounting
- AUTHORIZATION Peter;
- GO
- Создаем таблицу Invoices в схеме Accounting.
- CREATE TABLE Accounting.Invoices (
- InvoiceID int,
- InvoiceDate smalldatetime,
- ClientID int);
- GO
- Предоставляем разрешение SELECT на новую таблицу роли public.
- GRANT SELECT ON Accounting.Invoices
- TO public; GO
- Добавляем строку данных в новую таблицу.
- Обратите внимание на двухкомпонентное имя, которое мы используем для обращения к таблице в текущей базе данных.
- INSERT INTO Accounting.Invoices
- VALUES (101,getdate(),102);
- Удалить схему можно при помощи инструкции DROP SCHEMA. В SQL Server 2005 не допускается удаление схемы, если в схеме есть объекты. Информацию о схемах можно получить, выполнив запрос к представлению каталога sys.schemas. Следующий пример выполняет запрос к представлению каталога sys.schemas, чтобы получить информацию о схеме:
- SELECT *
- FROM sys.schemas;
- Следующий код (который можно найти в файлах примеров под именем ManagingAccessToSchemas02.sql) показывает, как удалить существующую схему, выполнив запрос к объектам, которые содержатся в этой схеме, и удалить сначала эти объекты.
- Изменяем контекст соединения на базу данных AdventureWorks.
- USE AdventureWorks;
- GO
- Извлекаем информацию о схеме Accounting.
- SELECT s.name AS "Schema",
- o.name AS "Object" FROM sys.schemas s INNER JOIN sys.objects o ON s.schema_id=o.schema_id WHERE s.name='Accounting'; GO
- Удаляем таблицу Invoices из схемы Accounting.
- DROP TABLE Accounting.Invoices;
- GO
- Удаляем схему Accounting.
- DROP SCHEMA Accounting;

Изменение прав доступа

- **Разрешения**
- **Описание**
- ALTER
- Разрешает изменять свойства таблицы
- CONTROL
- Предоставляет разрешения, аналогичные владению
- DELETE
- Разрешает удалять строки из таблицы
- INSERT
- Разрешает добавлять столбцы в таблицу
- REFERENCES
- Разрешает ссылаться на таблицу из внешнего ключа
- SELECT
- Разрешает осуществлять выборку строк из таблицы
- TAKE OWNERSHIP
- Разрешает присвоение схемы или таблицы
- UPDATE
- Разрешает изменять строки в таблице
- VIEW DEFINITION
- Разрешает доступ к метаданным таблицы