# ПОЛНОМОЧИЯ

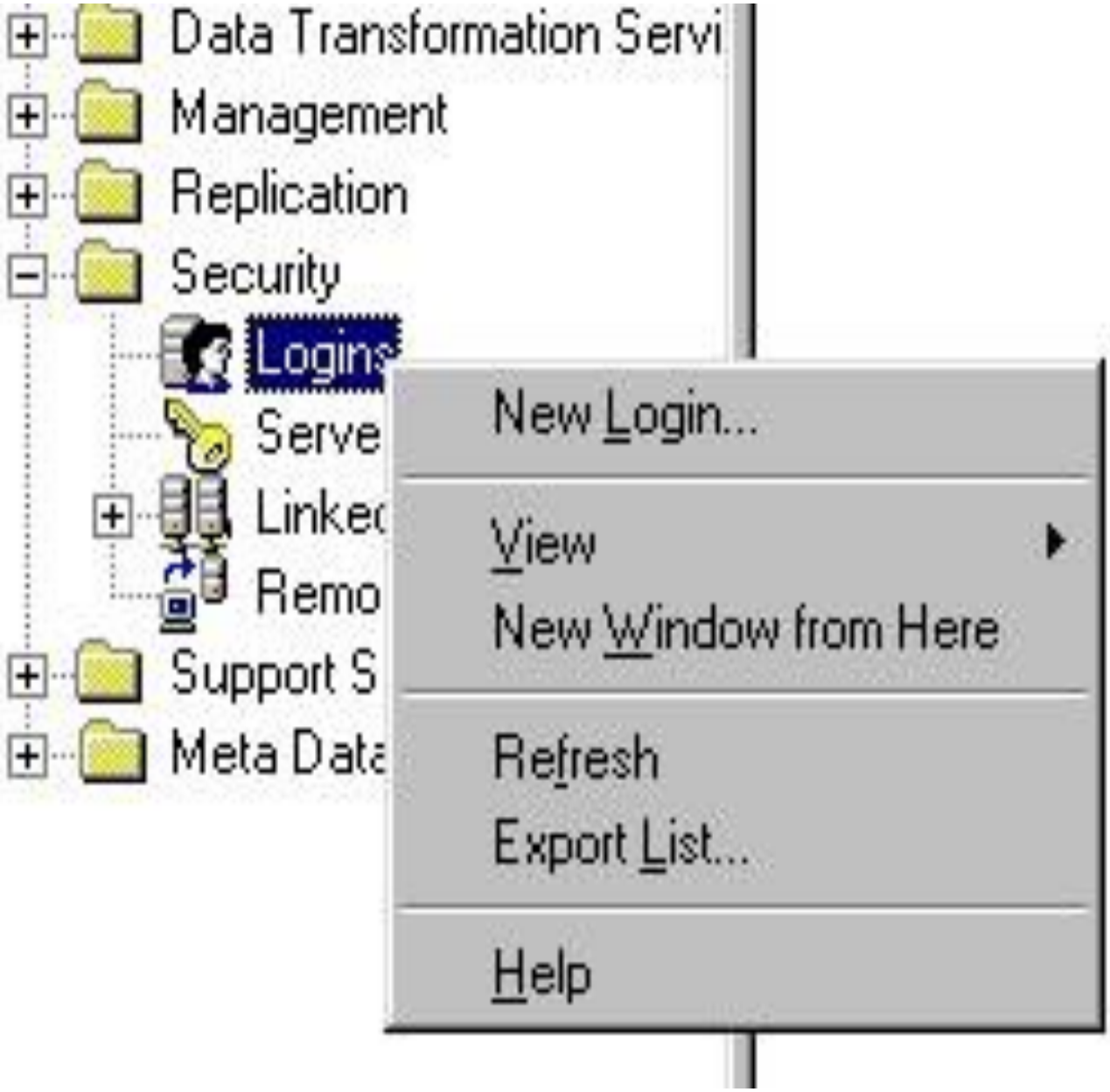
SQL Server располагает средствами, которые позволяют ограничить полномочия пользователей и приложений.

Например, можно разрешить одному пользователю только чтение таблицы, а другому разрешить все операции – чтение, вставку, удаление, модификацию.

# Регистрация пользователя

Для того, чтобы наделить пользователя какими-либо правами в базе данных, сначала следует зарегистрировать его на сервере, иначе он вообще не сможет установить соединение с сервером. Это можно сделать с помощью оболочки Enterprise Manager (EM).

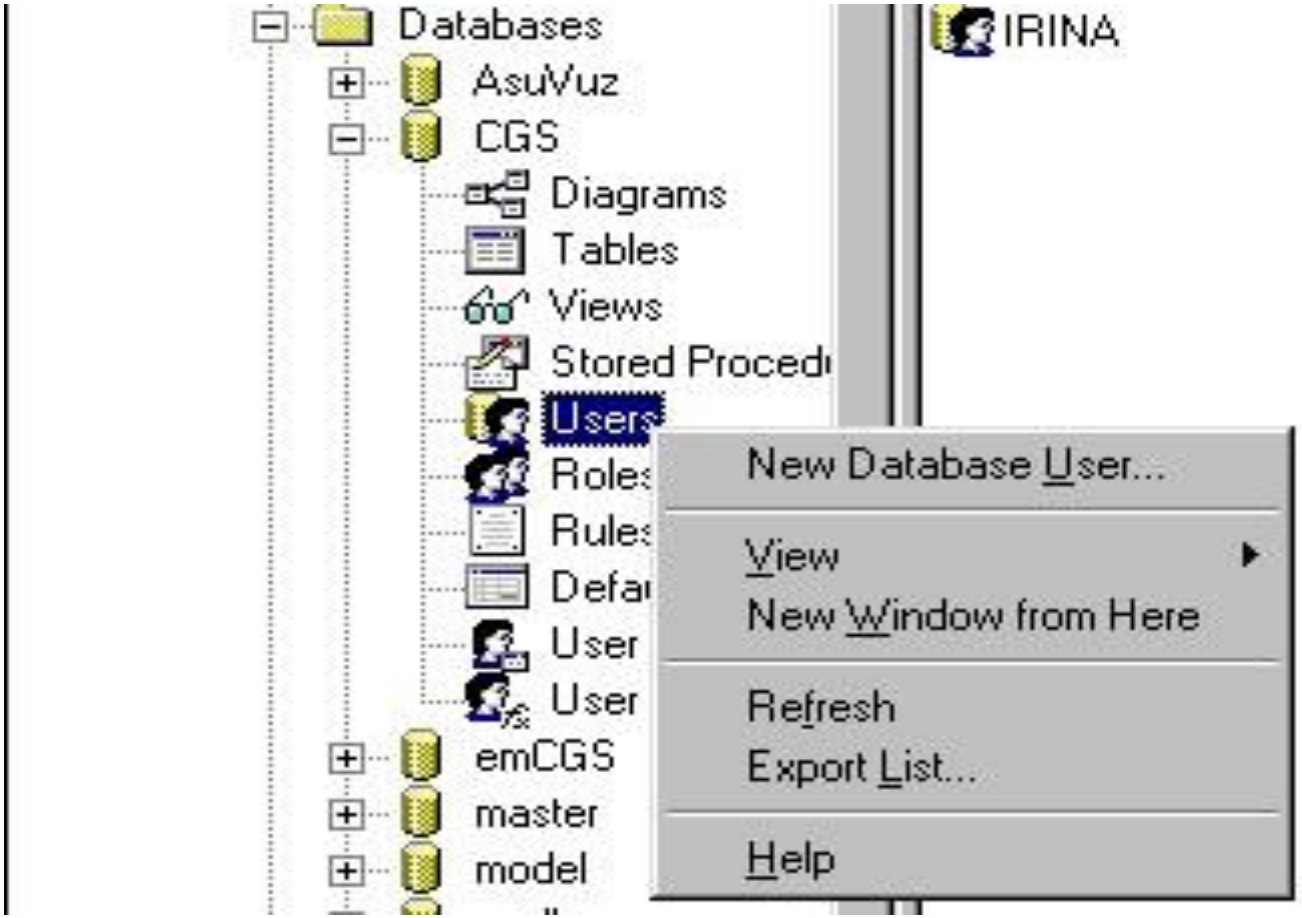
В консоли EM выберите сервер и раскройте узел Security:


Data Transformation Servi
Management
Replication
Security
Logins
Serve
Linked
Remo
Support S
Meta Data
New Login...
View
New Window from Here
Refresh
Export List...
Help

Выполните команду **New Login**

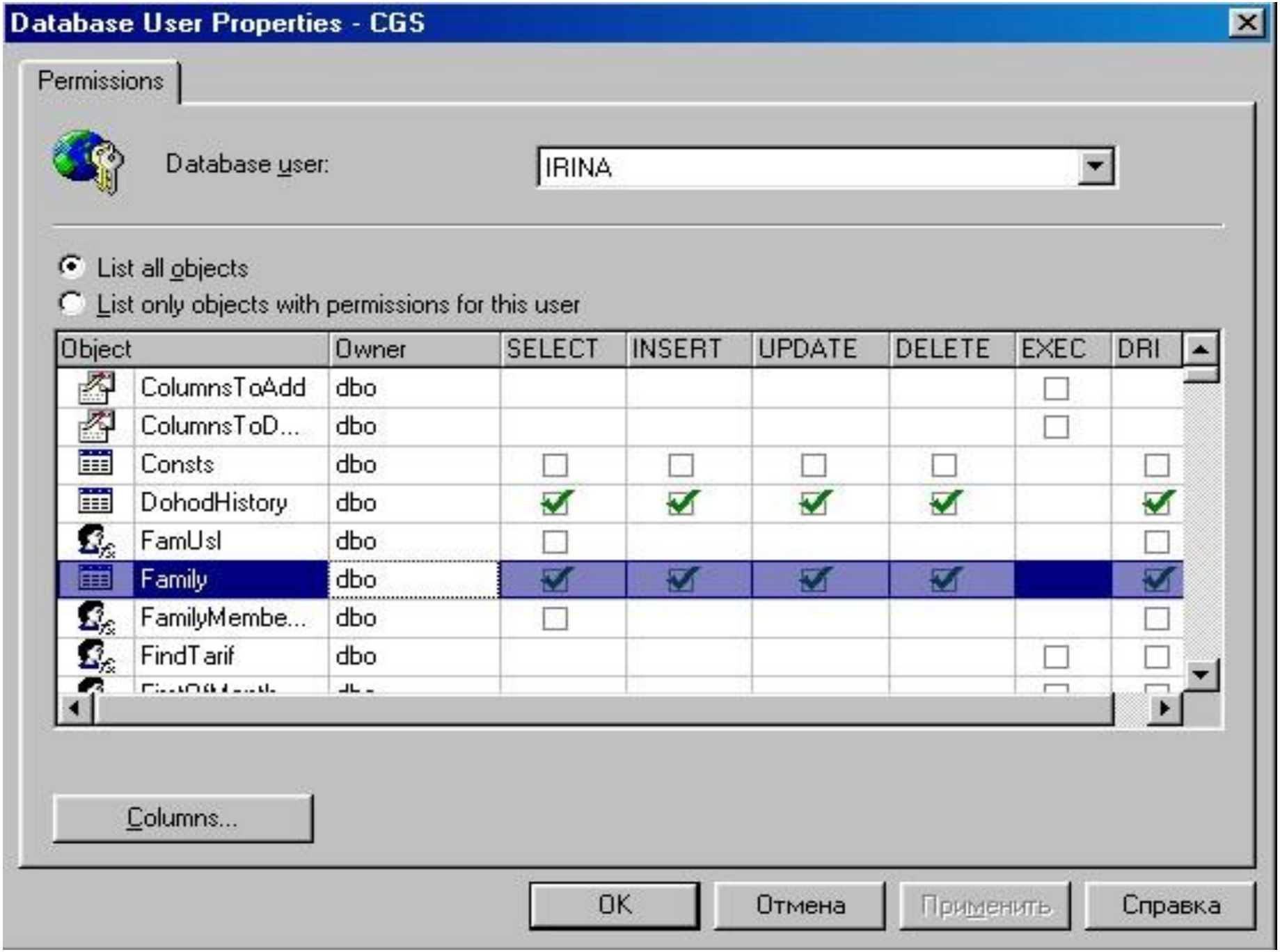
В диалоге укажите :имя, пароль и базу данных по умолчанию (необязательно), роли на сервере (если требуется), и отметьте те базы, к которым он вправе иметь доступ. Последнее можно сделать и позже.

Новый пользователь базы данных может быть зарегистрирован с помощью команды **New Database User**, доступной из контекстного меню узла **Users** консоли EM:



В раскрывающемся диалоге можно выбрать пользователя из числа пользователей, имеющих права на соединение с сервером и указать, принадлежащие ему роли.

Кнопка **Permissions** (разрешения) вызывает диалог, содержащий все объекты БД (таблицы, процедуры, функции, представления) и возможные операции с ними. Флажками следует отметить объекты и допустимые для пользователя действия над ними:

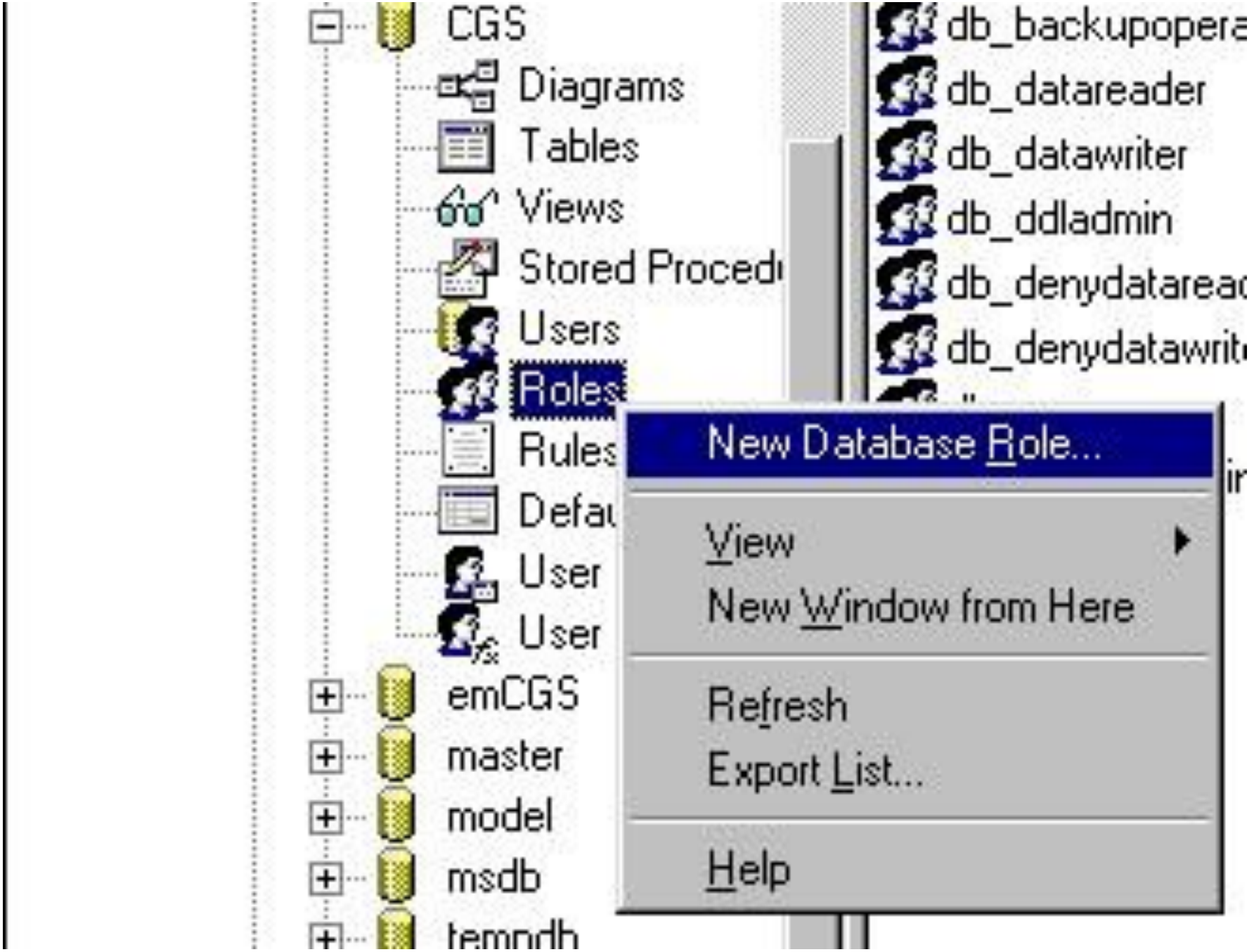

Database User Properties - CGS
Permissions
Database user:
IRINA
List all objects
List only objects with permissions for this user
Object
Owner
SELECT
INSERT
UPDATE
DELETE
EXEC
DRI
ColumnsToAdd
dbo
ColumnsToD...
dbo
Consts
dbo
DohodHistory
dbo
FamUsl
dbo
Family
dbo
FamilyMembe...
dbo
FindTarif
dbo
Columns...
OK
Отмена
Применить
Справка

# Роли

Как правило, не один, а группа пользователей БД должна обладать одними и теми же полномочиями.

Для того, чтобы не описывать заново полномочия нового пользователя каждый раз заново, можно создать роль, описать её полномочия, а затем придать эту роль пользователю.

Новая роль может быть создана в среде EM. Для создания новой роли выполните команду **New Database Role** из контекстного меню узла **Roles** в дереве консоли EM.


CGS
Diagrams
Tables
Views
Stored Proced
Users
Roles
Rules
Defau
User
User
emCGS
master
model
msdb
tempdb
db_backupopera
db_datareader
db_datawriter
db_ddladmin
db_denydatareac
db_denydatawrit
New Database Role...
View
New Window from Here
Refresh
Export List...
Help

В открывающемся диалоге создайте роль, а затем дайте ей соответствующие полномочия. Впоследствии вы сможете присвоить эту роль одному или более пользователей. Отметим, что пользователь может иметь более чем одну роль.

Роли бывают двух видов: стандартные роли и роли приложения. Стандартная роль создается процедурой

- **sp_addrole** [ **@rolename =** ] '*role*' [ **,** [ **@ownername =** ] '*owner*' ]

Существует предопределённая роль **PUBLIC** в любой базе данных. Ей могут быть даны произвольные полномочия, как и любой другой роли. Правами роли **PUBLIC** обладает любой пользователь, зарегистрированный в БД.

Ещё одна роль с фиксированными свойствами имеет имя **GUEST** (гость). Эта роль, в отличие от **PUBLIC** не создаётся автоматически.

Администратор БД должен создать её, если сочтёт нужным. Полномочиями этой роли обладают все пользователи, имеющие право на соединение с сервером, но не зарегистрированные в БД как её пользователи.

# Оператор GRANT

Оператор **GRANT** создает разрешение на доступ к объектам БД или выполнение операторов.

Оператор **GRANT**,дающий разрешение на выполнение операторов имеет синтаксис:

GRANT { ALL | *statement* [ **,**...*n* ] } TO *security_account* [ **,**...*n* ]

*Аргументы:*

ALL – дает все полномочия, которые вообще могут быть. Для разрешений на оператор ALL может быть использовано только членами роли **sysadmin**. Для разрешений на доступ к объектам - членами ролей **sysadmin** и **db_owner**, и владельцами объектов.

statement-оператор на выполнение которого даётся разрешение.

security account – имя пользователя, роли,

К операторам на выполнение которых даётся разрешение относятся:

CREATE DATABASE,

CREATE DEFAULT,

CREATE FUNCTION,

CREATE PROCEDURE,

CREATE RULE,

CREATE TABLE,

CREATE VIEW,

BACKUP DATABASE,

BACKUP LOG.

Операторы Transact-SQL, не требующие разрешений (их имеет роль **public**):

BEGIN TRANSACTION,

COMMIT TRANSACTION,

ROLLBACK TRANSACTION,

SAVE TRANSACTION,

RAISERROR,

PRINT,

SET.

Оператор GRANT, дающий разрешение на доступ к объектам:

```
GRANT { ALL [ PRIVILEGES ] | permission [ ,...n ] }
{
   [ ( column [ ,...n ] ) ] ON { table | view }
   | ON { table | view } [ ( column [ ,...n ] ) ]
   | ON { stored_procedure | extended_procedure }
   | ON { user_defined_function }
}
   TO security_account [ ,...n ]
   [ WITH GRANT OPTION ]
   [ AS { group | role } ]
```

## *Аргументы:*

ALL – дает все полномочия, которые вообще могут быть. ALL может быть использовано только членами роли **sysadmin**.

Для разрешений на доступ к объектам - членами ролей **sysadmin** и **db_owner**, и владельцами объектов.

*permission* – виды разрешений.

Для таблицы могут быть даны разрешения:

*select, insert, update, delete*.

Для таблицы может быть задан список столбцов, по отношению к которым действительно разрешение. Если список не задан, то разрешение относится ко всем столбцам таблицы. Для таблицы может быть также задано разрешение *references*, дающее право создавать ограничения *foreign key*.

Для хранимой процедуры и для функции существует разрешение *execute.*

*security account* – кому передаются полномочия. Это может быть пользователь БД, роль, пользователь Windows NT, группа Windows NT.

*with grant option* – означает, что пользователь, получивший полномочия, имеет право выполнить оператор *grant* для передачи их другим пользователям и/или ролям.

# Оператор DENY

Оператор запрещает пользователю, роли или группе наследовать полномочия благодаря членству в роли или группе.

Синтаксис оператора DENY для выполнения операторов:

```
DENY { ALL | statement [ ,...n ] } TO security_account [ ,...n ]
```

Пример:

```
DENY CREATE TABLE TO mk
```

Пусть mk является членом роли ROLE1. Теперь, если роль ROLE1 получит права на создание таблицы, пользователь mk этого права не получит.

Синтаксис оператора для обращения к объектам аналогичен синтаксису оператора GRANT.

```
DENY
  { ALL [ PRIVILEGES ] | permission [ ,...n ] }
{
  [ ( column [ ,...n ] ) ] ON { table | view }
  | ON { table | view } [ ( column [ ,...n ] ) ]
  | ON { stored_procedure | extended_procedure }
  | ON { user_defined_function }
}
  TO security_account [ ,...n ]
  [ CASCADE ]
```

CASCADE используется для каскадного запрета на получение полномочия(й).

Допустим, пользователь **A** имеет права на операции над таблицей **T.** Он передаёт это право пользователю **B**:

Grant select, update, insert, delete to B with grant option

Пользователь **B** в свою очередь передаёт это право пользователю **C**. Затем, некий член роли *db_owner* запрещает пользователю **A** удаление из этой таблицы.

```
deny delete on T to A
```

В результате, запрет распространится на пользователей **B** и **C**.

# Оператор *REVOKE*

отменяет данные ранее разрешения (grant) или запреты (deny).

Синтаксис для отмены прав на выполнение операторов:

```
REVOKE { ALL | statement [ ,..n ] } FROM
  security_account [ ,..n ]
```

# Синтаксис для отмены прав на операции доступа к объектам БД:

```
REVOKE [ GRANT OPTION FOR ]
    { ALL [ PRIVILEGES ] | permission [ ,...n ] }
    {
       [ ( column [ ,...n ] ) ] ON { table | view }
       | ON { table | view } [ ( column [ ,...n ] ) ]
       | ON { stored_procedure | extended_procedure }
       | ON { user_defined_function }
    }
  { TO | FROM }
    security_account [ ,...n ]
  [ CASCADE ]
```

# Аргумент GRANT OPTION FOR.

Употребление этого аргумента приводит к тому, что пользователь(и) сохраняют свои полномочия, но лишаются права их распространять.

Пример: отберём право на удаление записей из таблицы **T** у пользователя SomeUser.

```
REVOKE DELETE ON T TO SomeUser
```