Entendendo roles e suas permissões no ORACLE – grant e revoke

SEPTEMBER 9, 2011 [LEAVE A COMMENT](https://youoracle.wordpress.com/2011/09/09/entendendo-roles-e-permissoes-no-oracle-grant-e-revoke/#respond)

Você sabe como uma role funciona? Seu conceito é bastante simples: um agrupamento de permissões que pode ser concedida a usuários ou outras roles. Seu uso ajuda a administrar o permissionamento de objetos no banco, e poupa tempo que seria gasto com permissões e revogações individuais.

Vejamos um exemplo prático: no sistema de uma empresa, cada usuário deve ter acesso a determinadas tabelas e procedures, de acordo com seu nível hierárquico. Considere as tabelas ponto, horas\_trabalhadas e a procedure calcula\_total\_mensal.

Você precisa que o funcionário Mario possa gravar as horas\_trabalhadas e consulte a tabela ponto, mas não tenha acesso à soma das horas do mês. Para isso, executa os seguintes comandos.  
Exemplo:

SQL> grant insert on horas\_trabalhadas to mario;

Grant succeeded

SQL> grant select on ponto to mario;

Grant succeeded

SQL> revoke execute on calcula\_total\_mensal from mario

ORA-01927: cannot REVOKE privileges you did not grant

O erro ORA-01927: cannot REVOKE privileges you did not grant surge. Não se pode retirar uma permissão que nunca foi concedida. Se recebeu este erro, fique tranquilo, o usuário não possui esta permissão ao objeto e por isso não precisa ser revogado.

Dois dias se passam e outro funcionário é contratado, você precisa dar todas permissões novamente? E quais eram as permissões? O gerenciamento começa a ficar confuso. Vamos criar roles para gerenciar este cenário (para tal, é preciso ter o privilégio de sistema CREATE ROLE).  
Sintaxe:

create role role\_name [identified by] [using package] [externally] [globally];

* identified by | password – atribui uma senha para uso da role
* using | schema.package – será uma application role, somente aplicações poderão utilizá-la a partir de uma package especificada
* externally – o usuário deve ser autorizado por um serviço externo (so service ou third-party service) para habilitar a role
* globally – o usuário deve ser autorizado por um serviço de diretório para habilitar a role

Exemplo:

SQL> create role ro\_funcionario;

Role created

Agora daremos as permissões à esta role:  
Exemplo:

SQL> grant insert on horas\_trabalhadas to ro\_funcionario;

Grant succeeded

SQL> grant select on ponto to ro\_funcionario;

Grant succeeded

E criamos uma role para o nível de gerência, com acesso à procedure que soma o total de horas dos funcionários:  
Exemplo:

SQL> create role ro\_gerente;

Role created

SQL> grant execute on on calcula\_total\_mensal to ro\_gerente;

Grant succeeded

Agora, daremos à ro\_gerente permissão para fazer tudo o que um funcionário comum pode fazer, ou seja, atrubuiremos uma role à outra role:  
Exemplo:

SQL> grant ro\_funcionario to ro\_gerente;

Grant succeeded

Note que ainda não demos permissão efetiva a nenhum funcionário. Temos a ro\_funcionario que permite ações a funcionários comuns, e a ro\_gerente que além das permissões comuns, permite também visualizar relatórios mensais. Neste cenário, imagine que três pessoas foram admitidas: os funcionários Jose e Elias, e o gerente Pedro. Com as roles, podemos dar as permissões para cada um deles:  
Exemplo:

SQL> grant ro\_funcionario to jose;

Grant succeeded

SQL> grant ro\_funcionario to elias;

Grant succeeded

SQL> grant ro\_gerente to pedro;

Grant succeeded

Quando um novo funcionário for criado, basta dar permissão da role a ele.  
Caso precise remover da role a permissão de um objeto, basta usar o comando revoke para retirar o atributo previamente dado.

Apesar de ser um assunto simples, muitos bancos não fazem uso de diretivas de segurança com roles, o que torna o trabalho menos eficiente e mais suscetível a falhas de segurança. Em outro post falarei de algumas views do Oracle para descobrir quais roles estão atríbuidas a um usuário e quais permissões pertencem à uma role.