SubConsultas - Reforço

Objetivos:

- *Explorar as propriedades de subConsultas;*
- *Explorar propriedades de ordenação de informações;*
- *Atualizar informações de tabelas com uso de operações diversas;*
- *Converter dados de saída com uso do cast;*

Neste roteiro são explorados comandos para a realização de subconsultas, que são importantes para criar consultas avançadas e que possam sanar maiores necessidades de consultas SQL. Também são exploradas propriedades de ordenação e atualização de informações em tabelas que são úteis para a manutenção de bases de dados e tabelas.

Introdução

Algumas consultas necessitam que os valores existentes no banco de dados sejam buscados e depois utilizados em uma condição de comparação. Estas consultas podem ser formuladas com o uso de ***subconsultas*** *ou* ***consultas aninhadas****. É possível utilizar subconsultas quando desejamos buscar valores que não podem ser encontrados apenas com uma única consulta.*

Consultas Aninhadas

Este tipo de consulta pode ser implementada com o uso de blocos SELECT-FROM-WHERE completos dentro da cláusula WHERE de outra consulta. Uma maneira de utilizar uma consulta interna inicialmente é com o uso do operador IN, que compara um valor com um conjunto de valores ao mesmo tempo.

Exemplo básico de consulta com uso do operador IN:

```
SELECT f.tituloOriginal AS “Filme”, g.nome “Genero”
FROM filme f
INNER JOIN genero g ON f.idGenero = g.idGenero
WHERE
       g.nome IN('Suspense','Comédia',’Drama’,’Terror’,’Ação’)
ORDER BY g.idGenero;
```

Neste caso estamos procurando vários gêneros ao mesmo tempo, os que se adaptarem a nossa consulta serão retornados a tela. Mas a subconsulta, na verdade tem como objetivo principal reduzir o que estamos digitando e unificar uma pesquisa unindo duas consultas em uma só. Tudo o que temos a fazer é substituir uma parte da consulta anterior pelo que podemos chamar de subconsulta:

Exemplo melhor elaborado de consulta com operador IN:

```
SELECT f.tituloOriginal AS “Filme”, g.nome “Genero”
FROM filme f
INNER JOIN genero g ON f.idGenero = g.id
WHERE
       g.nome IN (SELECT genero.nome FROM genero)
ORDER BY g.id;
```

É possível ainda combinar vários tipos de consultas internas incluindo a maioria dos operadores utilizados em consultas e também mais de uma coluna, desde que sejam respeitadas a quantidade de colunas da cláusula e do resultante do que resultar do SELECT mais interno.

Operadores para subconsultas

São operadores que podem ser explicitados nas consultas com subconsultas e consultas onde os argumentos são ***relacionados****(consultas correlacionadas), a tabela a seguir descreve os operadores mais comuns que podem ser utilizados:*

Utilizando Operadores em SubConsultas			
Comando		***Função***	***Exemplo***
SELECT *campos* **FROM** *tabela* **WHERE** *argumentos*	**IN**	*Compara um valor v com um conjunto de valores V e avalia como TRUE se v for um dos elementos em V.*	**SELECT** f.tituloOriginal AS "Filme", g.nome "Genero" **FROM** filme f **INNER JOIN** genero g **ON** f.idGenero = g.id **WHERE** g.nome **IN**('Suspense','Comédia','Drama','Terror','Ação') **ORDER BY** g.id; *-- seleciona filmes apenas dos gêneros listados*
	EXISTS	*Verifica se o resultado de uma consulta aninhada correlacionada é vazio (não contém registros) ou não.* *Retorna TRUE se o resultado da consulta aninhada tiver ao menos um registro e FALSE se não houverem registros.*	**SELECT** filme.tituloOriginal **FROM** filme **WHERE EXISTS** (**SELECT** **NULL** **FROM** sessao **WHERE** sessao.idFilme = filme.idFilme); *-- seleciona os filmes que passaram ou irão passar em alguma sessão;*
OBSERVAÇÕES:		*É possível combinar os dois operadores de subconsultas com os demais operadores já utilizados (NOT, AND), e inclusive com o próprio IN ou EXISTS na mesma consulta.*	

Ordenando Informações

Em um banco de dados relacional, as linhas de uma tabela formam um grupo, não há ordem intrínseca entre as linhas e então temos de pedir ao MySQL para classificar os resultados se os desejarmos em uma ordem em particular. A tabela a seguir demonstra as maneiras de ordenar a informação recuperada em SELECTs.

Ordenando Listagens crescente e decrescente			
Alguns comandos utilizados em conjunto com a cláusula ORDER BY podem fazer com que a listagem retornada de uma consulta possa ser diferente do esperado, para tanto, existem operadores que podem ser combinados para alterar os resultados obtidos em listagens de dados.			
Comando		***Função***	***Exemplo***
SELECT *campos* ***FROM*** *tabela* ***WHERE*** *argumentos* ***ORDER BY*** *coluna*	*DESC*	*Classifica uma coluna em ordenação decrescente.*	***SELECT*** *nome* ***FROM*** *drinks* ***ORDER BY*** *nome* ***DESC;***
	ASC	*Classifica uma coluna em ordenação crescente, é o padrão de um ORDER BY quando não é informada nenhuma ordenação.*	***SELECT*** *nome* ***FROM*** *drinks* ***ORDER BY*** *nome* ***ASC;***

ATUALIZANDO DADOS EM TABELAS:

Os dados das tabelas existentes no banco de dados podem ser modificados caso necessite. Normalmente são tarefas rotineiras do banco de dados atualizas as informações existentes em nossas tabelas. Para isso, podemos utilizar o comando ***UPDATE*** *(atualizar).*

Atualizando dados em tabelas		
O UPDATE pode utilizar todos os parâmetros utilizados em consultas comuns no SQL criado. Podendo inclusive realizar a alteração de várias linhas de tabelas por vez.		
Comando	***Função***	***Exemplo***
`UPDATE tabela` `SET novo_valor` `WHERE condições`	*Altera dados de uma tabela mediante condições estipuladas*	`UPDATE drinks` `SET quantidade = 10` `WHERE igr_secundario LIKE '%abacaxi%'`

Ao atualizar informações de tabelas é possível combinar também diversas operações, envolvendo operadores de busca e também operações matemáticas nos valores das colunas, como o exemplo a seguir:

Exemplo de UPDATE envolvendo operações matemáticas

```
UPDATE filme
SET filme.duracao = filme.duracao + 30;
-- atualiza o tempo de todos os filmes cadastrados em mais 30;
```

DICA: Para estes tipos de operações podem ser combinados os operadores lógicos AND e OR;

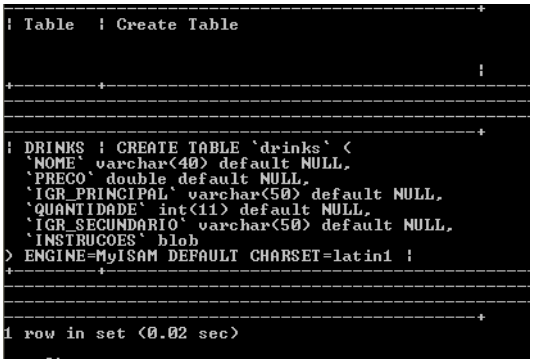
Forçando Classificações com CAST

A classificação da listagem de informações é sempre efetuada como apropriada para o tipo de coluna que está se utilizando. É possível forçar colunas para que comportem-se diferente utilizando a função ***CAST( )*** *seguida pela palavra chave* ***AS****. Vejamos as principais propriedades:*

Propriedade Cast			
Comando		***Função***	***Exemplo***
`SELECT campos` `FROM tabela` `WHERE argumentos` `ORDER BY` `CAST(coluna AS tipo)`	***AS CHAR***	*Para classificar como uma string de caractere.*	`SELECT nome` `FROM drinks` `ORDER BY CAST(nome AS CHAR);`
	AS SIGNED	*Para classificar como um número inteiro com sinal.*	`SELECT nome, preco` `FROM drinks` `ORDER BY CAST(preco AS SIGNED);`
	AS UNSIGNED	*Para classificar como um número inteiro sem sinal.*	`SELECT nome, preco` `FROM drinks` `ORDER BY CAST(preco AS UNSIGNED);`
	AS DATE	*Para classificar como uma data.*	`SELECT nome, preco` `FROM drinks` `ORDER BY CAST(preco AS DATE);`
	AS DATETIME	*Para classificar como uma data e hora.*	`SELECT nome, preco quant` `FROM drinks` `ORDER BY CAST(quant AS DATETIME);`
	AS TIME	*Para classificar como uma hora.*	`SELECT nome, preco quant` `FROM drinks` `ORDER BY CAST(quant AS TIME);`

VERIFICANDO COMO A ESTRUTURA DE SUAS TABELAS FOI CRIADA: *Podemos também desejar ver a estrutura das tabelas que foram criadas para testarmos como eles foram projetados. Para isso, podemos utilizar o comando SHOW organizado da seguinte forma:*

SHOW CREATE TABLE DRINKS;



Neste caso, podemos usar o comando para poder visualizar como os dados da sua tabela foram projetados e atendem aos requisitos para os quais foram projetados.

Tarefas

Use o banco de dados da Aula08 para desenvolver as questões a seguir.
Se for necessário crie os registros de inserção para testar as consultas solicitadas.

1) *Utilizando o **operador IN**, crie uma consulta para que **liste o nome** de todos os gêneros **menos os gêneros de suspense, terror e comédia**;*
2) *Utilizando **subconsultas**, crie uma consulta que **retorne os títulos, gênero e duração de filmes** em que o gênero seja **SUSPENSE e a duração** esteja **entre 70 e 130 minutos**;*
3) *Crie uma **function** incrementar **+44min** em **todos** os **filmes** de um determinado **gênero** (o gênero é passado como **parâmetro**).*
4) *Crie uma **procedure** que recebe o nome de um **ator** e informa os **filmes que ele participou**.*
5) *Crie uma **procedure** que informe o nome dos **atores** que **não participaram de nenhum filme;***
6) *Crie uma **procedure** que exiba o **título** e o **gênero** de todos os **filmes** que **não passaram** ainda em **cinema** algum;*
7) *Crie uma **procedure** que receba como parâmetro o nome de uma **cidade** e exiba os **cinemas existentes** nesta cidade.;*
8) *Crie uma **procedure** que receba dois parâmetros (**gênero atual**, **gênero novo**) e **altere** o gênero dos **filmes** de acordo com os parâmetros recebidos.*