EXPLORANDO QUERIES COM SQL

Atualizando informações no BD - UPDATE tabela SET
Removendo estruturas (complexas: tabela, database) no BD - DROP tabela ou nome BD
Removendo registros (para tuplas) - DELETE FROM
Modificar estrutura BD - ALTER TABLE tabela ADD ou ADD CONSTRAINT DROP COLUMN ou MODIFY COLUMN

Expressões
Eu posso ter como atributo do SELECT um atributo * outro. EX: salário * 0,1

Se eu tenho informações que vão derivar de uma informação base eu não preciso persisti-las.
Persistir dados simples e usar as operações (+,*,-) para trazê-los.

-> Concatenar informações no SELECT : concat (atributos) AS dá outro nome

round() - no MySQL eu escolho qtas casas decimais eu deixo após a vírgula

OPERADORES LÓGICOS

LIKE e BETWEEN
Like Cláusula para comparar

% - de 0 a vários elementos associados
_ - somente 1 caracter

Usar \ para definir que estamos usando String e não caracter especial

Between para intervalo: ex: (idade entre 10 e 80)

AND e OR

AND - quando todas as cláusulas devem ser verdadeiras
OR - uma ou outra deve ser verdadeira

OPERADORES MATEMÁTICOS

Union, Union All, Intersect (o que tem em comum em um conjunto e outro), Except

Só pode ser aplicado em tabelas que tenham a mesma estrutura, mesmo tipo de relação

NESTED QUERIES - ou SubQueries

Comparar valores recuperados e nesse caso o WHERE é necessário.
IN - comparação com conjuntos
Keywords: ANY, SOME, ALL - retornam true or false

Cláusulas EXISTS, NOT EXISTS, UNIQUE
Conjunto/comparação explícito - os valores estão explicitamente comparados e não recuperando-os

Um SubQuery interna enxerga um atributo ou cláusula de Querie externa, mas o inverso não. Se a tabela referenciada, por exemplo, estiver na subquerie você não consegue puxar um atributo dela na Query Principal.

QUERIES COM FUNÇÕES/CLÁUSULAS DE AGRUPAMENTO

Cláusulas de Ordenação (por colunas)

Order by - deixamos de ter conjuntos e passamos a ter listas - Pq uma lista tem uma sequência inerente a ela.
Usa um atributo de parâmetro para a ordenação.
Sentido/Ordem - Pode ser Ascendente (padrão) ASC ou Descendente DESC e pode usar mais de um atributo para ordenar.

Limit - extensão do SQL: exemplo ORDER BY atributo LIMIT 5

É possível ordenar atributos derivados e persistidos. Criar uma string (expressão) para ser o atributo do ORDER BY.

Funções de Agrupamento

Pode usá-las com SELECT e HAVING

COUNT - verificando somatório (contagem) de registros - LIDA COM COLUNAS
SUM - somatório dos valores de um atributo - LIDA COM VALORES PERSISTIDOS
MIN - valor mínimo entre os valores de um atributo - LIDA COM VALORES PERSISTIDOS
MAX - valor máximo entre os valores de um atributo - LIDA COM VALORES PERSISTIDOS
AVERAGE - média de valores de um atributo - LIDA COM VALORES PERSISTIDOS

Cláusula de Agrupamento

GROUP BY - agrupar por um atributo
Pode usar, por exemplo, um COUNT, um AVG e agrupar as informações com o GROUP BY

- Grupos de tuplas (grupo de valores). Cada grupo terá o mesmo valor de atributos, eles são comuns entre eles.
- Especifica os atributos de agrupamento

HAVING STATEMENT

Filtra os registros dentro dos grupos criados pelo GROUP BY.
Por exemplo: só vou agrupar determinados dados SE o Having for satisfeito.

Having é aplicado após o GROUP BY e antes do ORDER BY.
Having é uma condição de pode ser satisfeita (baseada no GROUP BY)

É importante entender a ordem de sequência/prioridade de execução.

REFORÇANDO DIFERENÇA **ORDER BY** E **GROUP BY**

Order By - ordem alfabética, por exemplo (pois quando inserimos os dados no BD não tem ordem nenhuma)

Group by - agrupa os registros iguais

CASE STATEMENT

Similar ao Switch-case em programação.
A partir de um bloco CASE - END eu posso aplicar vários WHEN - THEN - ELSE.
Posso usar o Case para Update, Select, para Query de agrupamento

- Vantagem de tornar a Query mais elaborada e conseguimos trocar valores a partir de uma condição de troca

Filtrar valores distintos em uma mesma query sem o Where.

JOIN

Faz uma junção entre as tabelas. A junção precisa retornar dados com coerência.
Após o ON utiliza-se onde (qual atributo) fará a junção das tabelas.
SELECT FROM JOIN ON WHERE

A JUNÇÃO é combinação e o WHERE é filtragem.

Cross Join (produto cartesiano) - Quando eu uso o Join sem o ON (sem o atributo de junção)
A cláusula CROSS JOIN **retorna todas as linhas das tabelas por cruzamento**, ou seja, para cada linha da tabela esquerda queremos todos os linhas da tabelas direita ou vice-versa

Se o atributo de Junção tem o mesmo nome em 2 tabelas, pode-se usar USING (nome atributo) no lugar de ON

INNER Join - tipo de Join mais utilizado
se não houver dado relacionado a junção, ele não vai aparecer (exclui o que não tiver match nas tabelas)

A ordem das tabelas nos JOIN's não altera o resultado.
Mas é possível forçar uma ordem = STRAIGHT_JOIN.

SELF JOIN

Utiliza a mesma tabela 2x na mesma consulta
Pq usar a mesma tabela em um JOIN? Juntar ela com ela mesma:

OUTER Join

OUTER JOIN - tudo da direita e tudo da esquerda
LEFT Join ou LEFT Outer Join - trago todos os dados mesmo que não tenham match entre as 2 tabelas - os resultados vem como NULL
RIGHT Join ou RIGHT Outer Join - recupera a interseção e toda a tabela da direita

NATURAL Join

Acontece entre 2 tabelas que tem uma junção implícita.
Há uma nomenclatura comum (mesmo nome de atributo) entre as duas tabelas