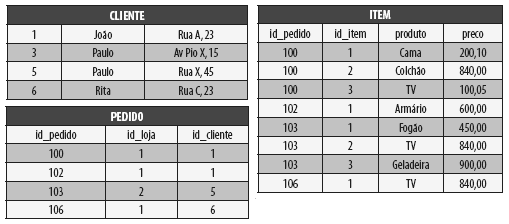
|  |  |  |  |
| --- | --- | --- | --- |
| **Universidade Luterana do Brasil**  **ULBRA – Campus Torres**  **Pró-Reitoria de Graduação**  rosa | | Avaliação Semestral | |
| Curso: STADS | Disciplina: Fundamentos de Bancos de Dados | | Ano/Semestre |
| Turma: quarta/quinta | Professores: Cássio Huggentobler Costa | | 2022-2 |
|  | | |

1. Atente as tabelas abaixo.



Pode-se afirmar que os relacionamentos entre as tabelas Cliente e Pedido e entre as tabelas Pedido e Item são, respectivamente: Letra D

1. 1:1 e 1:N
2. 1:N e 1:N
3. 1:N e 1:1
4. 1:N e N:N
5. N:N e 1:N

Marque a resposta certa e explique por que as outras estão erradas.

Letra A esta errada porque de acordo com a tabela pedido um cliente está em mais de um pedido, pela lógica um cliente pode ter vários pedidos, caso contrario ele poderia comprar apenas uma vez.

Letra B está errada porque um pedido pode ter vários itens assim como um item pode estar em vários pedidos  
  
Letra C está errada porque um item pode estar em vários pedidos diferentes

Letra E está errada pois um cliente pode ter vários pedidos porem um pedido só pode pertencer a um cliente

**Para a resolução dos exercícios subsequentes considere o esquema de banco de dados, nomeado “Acadêmico“, descrito abaixo:**

Professor (idprof, nome, email, endereço, cpf, idcid);

idcid referencia cidade(idcid)

Aluno (idaluno, nome, endereço, telefone, idcid)

idcid referencia cidade(idcid)

Cidade(idcid, nome, UF)

Turma(idturma, idprof, sala, capacidade, idcurso)

idcurso referencia Curso (idcurso)

idprof referencia Professor (idprof)

Disciplina(iddidisc, nome)

Turdisc(idturma, iddisc)

idturma referencia Turma(idturma)

iddisc referencia Disciplina(iddisc)

Matricula(idaluno, idturma)

idaluno referencia Aluno (idaluno)

idturma referencia turma (idturma)

Curso (idcurso, nomecurso, valorcredito)

2. Com base no esquema “Acadêmico“, analise as seguintes afirmações;

I - A tabela Aluno precisa ser criada antes da tabela Disciplina,

II - Antes de criar a tabela Turdisc, as tabelas Turma e Disciplina devem estar criadas.

III - A tabela Curso pode ser a última a ser criada,

É correto somente o que se afirma em:

1. I
2. II
3. III
4. I e II
5. II e III

Explique por que as outras estão erradas.

Letra C esta errada pois a tabela curso está sendo referenciada em outras tabelas, portanto não pode ser a ultima a ser criada

Letra A está errada porque na tabela aluno

não utiliza uma foreign key da tabela disciplina, logo não precisa ser criada antes.

3. Marque a sentença que permite listar todas os nomes dos alunos juntamente com o nome da cidade onde moram, ordenado pelo nome do aluno.

1. (\_\_\_) Select aluno.nome, cidade.nome from cidade, aluno
2. (\_\_\_) Select aluno.nome, cidade.nome from aluno,cidade where cidade.idcid=aluno.idaluno order by aluno.nome
3. (\_\_\_) Select aluno.nome, cidade.nome from cidade, aluno where cidade.nome=aluno.nome order by aluno.nome
4. (\_\_\_) Select aluno.nome, cidade.nome from cidade, aluno where cidade.idcid=aluno.idcid
5. (\_X\_) Select aluno.nome, cidade.nome from cidade, aluno where cidade.idcid=aluno.idcid order by aluno.nome

* Da resposta que você selecionou correta, refatore-a utilizando joins.

Versão PostgresSQL  
SELECT aluno.nome AS "Nome do aluno", cidade.nome AS "Nome da cidade"

FROM aluno

JOIN cidade ON cidade.idcid=aluno.idcid

ORDER BY aluno.nome

4. Faça um SQL que permite atualizar o valor do crédito do curso de nome “ADS” com o acréscimo de 10%.

Mostre o SQL que consulta também, antes de atualizar os dados.

SELECT credito

FROM cursos

WHERE nome = 'ADS';

SELECT ROUND(SUM(credito + credito\*10/100),2) AS "Crédito com 10% de acréscimo"

FROM cursos

WHERE nome = 'ADS';

5. Marque a sentença que permite listar os códigos(id) das turmas com o nome da disciplina e a sala.

1. (\_x\_) Select idturma, nome, sala from turma, turdisc, disciplina where turma.idturma=turdisc.idturma and disciplina.iddisc=turdisc.iddisc
2. (\_\_\_) Select idturma, nome, sala from turma
3. (\_\_\_) Select idturma, nome, sala from turma, disciplina where turma.idturma=disciplina.iddisc
4. (\_\_\_) Select idturma, nome, sala from turma, disciplina where disciplina.iddisc=turdisc.iddisc
5. (\_\_\_) Select codturma, nome, sala from turma, disciplina, turdisc

* refatore a resposta escolhida como certa, para utilizar JOINS.

6. Defina o comando SQL para listar o nome do curso e valor do crédito para o curso que tem o valor mais baixo de crédito(valorcredito).

1. (\_\_\_) Select nomecurso, min(valorcredito) from curso
2. (\_\_\_) Select nomecurso, valorcredito from curso where valorcredito=(select min(valorcredito) from curso)
3. (\_\_\_) Select nomecurso, max(valorcredito) from curso
4. (\_\_\_) Select nomecurso, valorcredito from curso where min(curso)
5. (\_\_\_)Select nomecurso, min(valorcredito) from aluno where min(valorcredito)

* refatore a resposta escolhida como certa, para utilizar JOINS.

7. Explique o conceito Integridade Referencial. De um exemplo SQL DDL para ilustrar melhor a sua explicação, usando a modelagem relacional (“Academico”).

8. Faça um SQL para adicionar a coluna data\_entrada na tabela matricula.

ALTER TABLE matricula

ADD COLUMN data\_entrada DATE;

9. Faça um SQL para selecionar os alunos que não moram na cidade onde estudam. Considerando que a cidade onde estudam é Torres. Exibir o código, o nome do aluno e a cidade onde ele mora.

SELECT aluno.id AS "Código do aluno",

aluno.nome AS "Nome",

cidade.nome AS "Cidade"

FROM aluno

join cidade ON aluno.idcid = cidade.idcid

where cidade.nome != 'Torres';

10. Faça um SQL para exibir o nome do aluno, o código do curso, o nome do curso e o nome do professor da turma e a data que ele se matriculou. Apresentar somente para alunos que tenham se matriculado entre semestre 1 de 2022, considere que as matrículas iniciam em janeiro.

SELECT aluno.nome AS "Aluno",

curso.id AS "Código do curso",

curso.nome AS "Nome do curso",

professor.nome AS "Nome do professor",

aluno.data\_matricula AS "Data da matricula"

FROM aluno

JOIN curso ON aluno.id\_curso = curso.id

JOIN professor ON curso.id\_professor = professor.id

WHERE aluno.data\_matricula BETWEEN '2022-01-01' AND '2022-06-01';

11. Faça um SQL que liste os 2 professores que mais turmas já atenderam. Importante mostrar o nome do professor.

SELECT professor.nome,

COUNT(turma.id) AS "Quantidade de turmas"

FROM professor

JOIN turma ON professor.id = turma.id\_professor

GROUP BY professor.nome

ORDER BY 2 DESC

LIMIT 2;