UNIDADE 5 – INTRODUÇÃO AO JPA

1. [CESPE - 2013 - CNJ] Os objetos mapeados na linguagem Java que devem ser persistidos como objetos precisam utilizar JPA (Java Persistence API), pois o JPA permite realizar o mapeamento objeto/relacional automatizado e transparente e sua persistência em um banco de dados relacional.

(a) Certo (b) Errado

2) [CESPE - 2014 - TJ-SE] Julgue o item abaixo, relativo à JPA (Java Persistence API).
A JPA, que foi criada como alternativa para o Hibernate para conexão com os sistemas gerenciadores de banco de dados, está nativa no Java SE a partir da versão 1.3.

(a) Certo (b) Errado

3. [FCC - 2014 - TRT - 13ª Região (PB)] Java Persistence API (JPA) é uma API padrão da linguagem Java para persistência de dados em bancos de dados relacionais. Em uma aplicação que utiliza JPA

(a) pode ser utilizada, como provedor de persistência, as bibliotecas EclipseLink, Hibernate, OracleTopLink, JBossSeam e JDBCProvider.

(b) as classes de entidade do banco de dados permitem o mapeamento entre objetos da classe e tabelas do banco de dados, utilizando anotações como @Table, @Entity, @PrimaryKey, @Column, @Constraint, @Foreignkey e @EJB.

(c) todas as operações realizadas nas tabelas do banco de dados, como inserção de dados, alteração, consultas e exclusão, são realizadas sem o uso de instruções SQL, ou seja, o desenvolvedor não precisa conhecer SQL para programar.

(d) as configurações de acesso a banco de dados normalmente ficam no arquivo persistence.xml, ligado à aplicação, de forma que se for alterado o servidor de banco de dados não seja necessário alterar o código-fonte Java da aplicação.

(e) as relações existentes entre as tabelas do banco de dados não são refletidas nas classes de entidade criadas na aplicação, o que torna a execução mais rápida. O mapeamento de relações é feito em tempo de execução pelas bibliotecas do provedor de persistência.

4. [FCC - 2011 - TRT - 23ª REGIÃO (MT)] Considere:

```
@Entity
@Table(name = "domic")
@NamedQueries ({
@NamedQuery(name = "Domic.findById", query = "SELECT r FROM Domic r WHERE r.id = :id"),
@NamedQuery(name = "Domic.findByNome", query = "SELECT r FROM Domic r WHERE r.nome = :nome")
})
public class Domic implements Serializable {
   private static final long serialVersionUID = 1L;
   @Id
   @Column(name = "id", nullable = false)
   private Integer id;
   @Column(name = "nome")
   private String nome;
   @OneToMany(cascade = CascadeType.ALL, mappedBy = "domicId")
   private Collection<Predio> predioCollection;
```

Em relação à JPA (Java Persistence API) é INCORRETO afirmar que

(a) @NamedQuery é aplicada para definir várias consultas.
(b) @Entity define que haverá correspondência da classe com uma tabela do banco de dados.
(c) @Id define que o atributo que está mapeado com tal anotação corresponderá à chave primária da tabela.
(d) @Column(name = "id", nullable = false) define que o atributo da classe mapeado com tal anotação deve estar associado à coluna cujo nome é "id", além de definir que tal campo não pode ser nulo.
(e) @OneToMany indica que o atributo contém um conjunto de entidades que a referenciam.

5. [CESPE - 2010 - TRE-BA] As tecnologias JPA e EJB permitem, com o uso da linguagem Java, a manipulação de dados que estão em um banco de dados.

(a) Certo (b) Errado

6. [FUMARC - 2011 - BDMG] Analise as seguintes afirmativas sobre fundamentos de servidores de

aplicação.

I. Um servidor de aplicação disponibiliza um ambiente para a instalação e execução de determinadas aplicações. Os servidores de aplicação web também são conhecidos como middleware.
II. JPA é uma API que padroniza o acesso a banco de dados através de mapeamento Objeto/Relacional dos Enterprise Java Beans.
III. JTA é uma API que padroniza o tratamento de transações dentro de uma aplicação Java.

Marque a alternativa CORRETA:

(a) apenas as afirmativas I e II são verdadeiras.
(b) apenas as afirmativas I e III são verdadeiras.
(c) apenas as afirmativas II e III são verdadeiras.
(d) todas as afirmativas são verdadeiras.

7) [FCC - 2011 - TCE-PR] A JPA

(a) pode ser usada fora de componentes EJB e fora da plataforma Java EE, em aplicações Java SE.

(b) utiliza persistência gerenciada por contêiner (CMP), ou seja, as classes de entidade e persistência necessitam de um contêiner presente em um servidor de aplicações para serem executadas.

(c) utiliza descritores XML para especificar informações do mapeamento relacional de objeto, mas não oferece suporte a anotações.

(d) suporta consultas dinâmicas nomeadas nas classes de entidade que são acessadas apenas por instruções SQL nativas.

(e) possui uma interface EntityBeans que padroniza operações Createpdate Delete (CRUD) que envolvem tabelas.

8. [FCC - 2012 - TJ-PE] Quando se utiliza JPA, um EntityManager mapeia um conjunto de classes a um banco de dados particular. Este conjunto de classes, definido em um arquivo chamado persistence.xml, é denominado

(a) persistence context. (b) persistence unit.

(c) entity manager factory. (d) entity transaction.

(e) persistence provider.

9. [FCC - 2012 - TJ-PE] Em uma classe de entidade de uma aplicação que utiliza JPA, a anotação que define um atributo que não será salvo no banco de dados é a

(a) @Optional. (b) @Transient. (c) @Stateless.

(d) @Stateful. (e) @Local.

10. [FCC - 2012 - TCE-AM] Considere o excerto a seguir:

Em uma aplicação que utiliza Hibernate, uma I representa uma determinada configuração de repositório de dados (data-store) lógicos. A I I tem o mesmo papel em uma aplicação JPA, e configura-se uma II I com arquivos de configuração ou em código da aplicação assim como se configuraria uma IV . A configuração de uma V , junto com um conjunto de metadados de mapeamento (normalmente classes anotadas), é chamada de VI .

As lacunas I, II, III, IV, V e VI devem ser preenchidas, correta e respectivamente, por:

(a) Java Transaction API - EntityManager - EntityManagerFactory - EntityTransaction - Java Transaction API - persistence unit.

(b) EntityManager - EntityManager - EntityManagerFactory - EntityTransaction - SessionFactory - driver.

(c) Connection - DriverManager - DriverManager - Connection - Connection - statement.

(d) EntityTransaction - Connection - Connection - SessionFactory - Java Transaction API - persistence unit.

(e) SessionFactory - EntityManagerFactory - EntityManagerFactory - SessionFactory - EntityManagerFactory - unidade de persistência.

GABARITO

1 – A; 2 – B; 3 – D; 4 – A; 5 – A; 6 – D; 7 – A; 8 – B; 9 – B; 10 - E