Sistemas Informáticos I

Control Intermedio 2

29 de noviembre de 2018

Test	1	2	3	4	5	6	Total

Apellidos:

Nombre:

Test (1 punto): 4 preguntas. Cada respuesta correcta suma 0,25 puntos, incorrectas restan 0,1 puntos.

1. ¿Cuál de estas afirmaciones es verdadera?
   a) Todo procedimiento almacenado es un trigger.
   b) El lenguaje PL/pgSQL permite crear funciones que se ejecutan en el gestor de base de datos PostgresSQL.
   c) En un procedimiento almacenado, el resultado de cualquier consulta se puede almacenar en una variable utilizando SELECT INTO.
   d) En un trigger, el uso de BEFORE y AFTER aplicado a INSERT tienen el mismo resultado.

2. Suponga que la tabla persona tiene definido un campo nombre y el siguiente código de creación de un trigger:
CREATE TRIGGER cambio BEFORE INSERT OR UPDATE ON persona FOR EACH ROW EXECUTE PROCEDURE cambio ();
   a) Dentro del código de cambio() la expresión NEW.nombre será siempre NULL
   b) Dentro del código de cambio() OLD.nombre será siempre NULL
   c) El retorno de cambio()será ignorado siempre
   d) El retorno de cambio()puede alterar la inserción o modificación de una fila en persona

3. ¿Cuál de las siguientes afirmaciones sobre Hadoop es verdadera?
   a) En Hadoop, la suma total de capacidad de proceso y almacenamiento es igual a la suma de capacidad de proceso y almacenamiento de cada uno de sus nodos.
   b) Aunque continuamente surgen alternativas tecnológicas, se prevé que Hadoop siga siendo el referente de Big Data por varios años.
   c) La principal limitación de Hadoop es que para su ejecución requiere que estén disponibles grandes cantidades de memoria RAM en cada nodo.
   d) La principal limitación de Hadoop es que es incompatible con la técnica MapReduce, que es la técnica preferida en la actualidad.

4. ¿Cuál de estas afirmaciones sobre el teorema del CAP es falsa?
   a) Implica que a la hora de elegir un gesto de base de datos se debe renunciar a la Consistencia (permanente), a la Disponibilidad (total para todos los nodos) o a la Tolerancia al Particionamiento.
   b) Establece que un gestor de base de datos, para servir en aplicaciones Big Data, debe satisfacer las condiciones de Consistencias, Disponibilidad y Tolerancia al Particionamiento (fallos aislantes).
   c) En un gestor de base de datos apto para tratamiento de BigData no tiene sentido pensar en dar soporte sólo a la Consistencia y Disponibilidad.
   d) Un gestor de base de datos que implemente las condiciones de Disponibilidad y Tolerancia al Particionamiento, forzosamente debe permitir escrituras inconsistentes.

1- (2,5 puntos) En una base de datos SQL existe una tabla *Dispositivo* con dos campos, *marca* y *modelo*, de tipo cadena de caracteres. Un posible estado de esta tabla podría ser:

marca	modelo
Apple	iPhone 7
Samsung	Galaxy S7
Sony	Xperia X
Apple	iPhone 6

Suponiendo que la base de datos **NO** está vacía y que se debe mantener la coherencia de los datos, escribir las sentencias SQL necesarias para modificar la estructura de la tabla *Dispositivo* para que en lugar de un campo de texto con la marca incluya una referencia a una nueva tabla con todas las posibles marcas. Por ejemplo, suponiendo que el estado de la tabla es el anterior, después de ejecutar las sentencias SQL la base de datos contendría esta información:

DISPOSITIVO

idMarca	modelo
1	iPhone 7
2	Galaxy S7
3	Xperia X
1	iPhone 6

MARCA_DISPOSITIVO

id	marca
1	Apple
2	Samsung
3	Sony

2- (1 punto) ¿Cuál es la diferencia entre un DataSource físico y uno lógico? Explica detalladamente los principales beneficios que aporta utilizar DataSource lógicos en un sistema informático con una arquitectura cliente-servidor.

MyBatis se puede definir como un framework de persistencia que soporta SQL, procedimientos almacenados y mapeos avanzados. MyBatis elimina casi todo el código JDBC, el establecimiento manual de los parámetros y la obtención de resultados. En el contexto de esta definición:

3- (0.5 puntos) ¿Por qué se considera un framework de persistencia que elimina casi todo el código JDBC?

4- (1 punto) ¿Que se entiende por "mapeo"? ¿Por qué se considera que los ofrecidos por MyBatis son "avanzados"?

[4]

Para los siguientes ejercicios considere las tablas *actor, film_actor, film, film_category y category*, cuyo código SQL de creación se muestra a continuación.

```
CREATE TABLE actor (
    actor_id integer NOT NULL DEFAULT nextval('actor_actor_id_seq'::regclass),
    first_name character varying(45) NOT NULL,
    last_name character varying(45) NOT NULL,
    last_update timestamp without time zone NOT NULL DEFAULT now(),
    CONSTRAINT actor_pkey PRIMARY KEY (actor_id)
);

CREATE TABLE film_actor (
    actor_id integer NOT NULL,
    film_id integer NOT NULL,
    last_update timestamp without time zone NOT NULL DEFAULT now(),
    CONSTRAINT film_actor_pkey PRIMARY KEY (actor_id, film_id),
    CONSTRAINT film_actor_actor_id_fkey FOREIGN KEY (actor_id)
        REFERENCES public.actor (actor_id) MATCH SIMPLE,
    CONSTRAINT film_actor_film_id_fkey FOREIGN KEY (film_id)
        REFERENCES public.film (film_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE RESTRICT
);

CREATE TABLE film (
    film_id integer NOT NULL DEFAULT nextval('film_film_id_seq'::regclass),
    title character varying(255) NOT NULL,
    description text,
    release_year year,
    last_update timestamp without time zone NOT NULL DEFAULT now(),
    CONSTRAINT film_pkey PRIMARY KEY (film_id)
);

CREATE TABLE film_category (
    film_id integer NOT NULL,
    category_id integer NOT NULL,
    last_update timestamp without time zone NOT NULL DEFAULT now(),
    CONSTRAINT film_category_pkey PRIMARY KEY (film_id, category_id),
    CONSTRAINT film_category_category_id_fkey FOREIGN KEY (category_id)
        REFERENCES public.category (category_id) MATCH SIMPLE,
    CONSTRAINT film_category_film_id_fkey FOREIGN KEY (film_id)
        REFERENCES public.film (film_id) MATCH SIMPLE
);

CREATE TABLE category (
    category_id integer NOT NULL DEFAULT nextval('category_category_id_seq'::regclass),
    name character varying(25) COLLATE pg_catalog."default" NOT NULL,
    last_update timestamp without time zone NOT NULL DEFAULT now(),
    CONSTRAINT category_pkey PRIMARY KEY (category_id)
)
```

5- (2 puntos). Escriba una consulta SQL que devuelva los 10 actores que más participaciones tengan en películas de la categoría “Action”.

6- (2 puntos). Suponga que ya existe una tabla creada con el siguiente comando:

```
CREATE TABLE total_reparto (
    peli_id integer NOT NULL,
    total integer DEFAULT 0,
    CONSTRAINT total_reparto_pkey PRIMARY KEY (peli_id),
    CONSTRAINT total_reparto_peli_id_fkey FOREIGN KEY (peli_id)
        REFERENCES public.film (film_id) MATCH SIMPLE
);
```

Escribir el código de creación en PostgreSQL de un trigger que cada vez que se intente insertar un nuevo registro en la tabla *film_actor* verifique que la película correspondiente no tenga más de 20 actores, en cuyo caso la inserción no debe hacerse. Además, el trigger debe mantener actualizada la tabla *total_reparto*.