|  |
| --- |
| **Evaluación y Prueba Técnica: Oracle y PostgreSQL** |

**Evaluación Técnica:**

1. Explique ¿qué es un cursor en ORACLE? y escriba un pequeño ejemplo de cómo declarar y leer un cursor.

Es un objeto el cual nos permite recorrer los resultados de forma secuencial, los cursores se necesitan para procesar las filas de una tabla como por ejemplo una tabla de grandes registros.

EJEMPLO:

DECLARE

CURSOR c\_carro IS

select nombre

FROM carros

where marca = ‘TOYOTA’;

CURSOR:

BEGIN

FOR carros\_name IN c\_carro LOOP

DBMS\_OUTPUT.PUT\_LINE(‘Nombre del carro: ’ || carros\_name.nombre);

END LOOP;

END;

1. Seleccione la opción correcta:
2. NO es atributo de un cursor:
3. %ROWCOUNT
4. %FOUND
5. %NOTFOUND
6. %ISOPEN
7. %ISVALID
8. Para qué sirve la sentencia PIVOT:
9. Permite trasponer la salida de una consulta de filas a columnas.
10. Permite trasponer la salida de una consulta de columnas a filas.
11. Obtiene el valor promedio de los valores de un campo específico.
12. Sirve para poder combinar las operaciones de insert y update según una condición dada.
13. Ninguna de las anteriores.
14. Indique cual es la afirmación incorrecta sobre la función GENERATE\_SERIES():
15. Recibe como máximo 3 parámetros.
16. Es una función nativa exclusiva de Oracle.
17. Se usa para devolver una serie de valores entre un punto de inicio y uno de final.
18. Es una función nativa exclusiva de PostgreSQL.
19. Recibe como primer parámetro el valor inicial de la serie a generar.
20. Indique la afirmación correcta sobre la función CROSSTAB():
21. Función que viene incorporada en la extensión tablefunc.
22. Es una función nativa de PostgreSQL.
23. Función que genera una tabla de referencia cruzada con columnas especificadas por una segunda consulta.
24. Todas las anteriores son correctas.
25. Solo a y c son correctas.
26. ¿Qué son las extensiones en PostgreSQL y cómo se pueden utilizar para extender la funcionalidad del sistema?

Son módulos adicionales que nos da la facilidad de agregar nuevas funcionalidades a la bd de forma sencilla y organizada.

Se implementan mediante archivos .so en Unix y .dll en Windows.

Instalar la extensión, cargar la extensión y utilizar la funcionalidad.

**Prueba Técnica:**

En esta prueba técnica, se solicita crear procedimientos almacenados que cumplan con los requerimientos solicitados, se deben crear los procedimientos almacenados tanto para una base de datos en Oracle como en PostgreSQL. Crear los procedimientos sobre las entidades indicadas:

**Tabla de Coberturas:**

|  |  |  |
| --- | --- | --- |
| **ID\_Cobertura** | **Nombre\_Cobertura** | **Descripcion** |
| 1 | Seguro de Vida | Cobertura que protege en caso de fallecimiento. |
| 2 | Seguro de Salud | Cobertura médica para gastos de salud. |
| 3 | Seguro de Automóvil | Cobertura para daños a vehículos y terceros. |

**Tabla de Seguros:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID\_Seguro** | **ID\_Cobertura** | **Nombre\_Seguro** | **Descripcion** | **Precio** | **Moneda** | **Duracion** |
| 1 | 1 | Vida Total | Cobertura completa de seguro de vida. | 500 | PEN | Anual |
| 2 | 2 | Salud Básico | Cobertura médica básica. | 200 | USD | Mensual |
| 3 | 3 | Auto Terceros | Cobertura para daños a terceros. | 300 | PEN | Anual |

**Tabla de Clientes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID\_Cliente** | **Nombre\_Cliente** | **Direccion** | **Correo\_Electronico** | **Telefono** | **Fecha\_Nacimiento** | **Sexo** |
| 1 | Juan Pérez | Calle 123, Lima | juan.perez@example.com | 999888777 | 1985-03-15 | M |
| 2 | María Gómez | Jr. Paz 451, Tacna | maria.gomez@example.com | 999999888 | 1990-11-22 | F |
| 3 | Carlos López | Calle 8, Ica | carlos.lopez@example.com | 999888999 | 1988-07-10 | M |
| 4 | Marcos Flores | Jr. San Martin, Lima | marcos.f26@example.com | 999888990 | 1988-06-15 | M |
| 5 | Vilma Sanchez | Jr Junin, Ayacucho | sanchez.vilma@example.com | 999128999 | 1988-07-10 | F |

**Tabla de Pólizas:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID\_Poliza** | **ID\_Cliente** | **ID\_Seguro** | **Fecha\_Inicio** | **Fecha\_Vencimiento** | **Estado de la Póliza** |
| 1 | 1 | 1 | 2023-01-01 | 2024-01-01 | Activo |
| 2 | 2 | 2 | 2023-02-15 | 2023-03-15 | Activo |
| 3 | 3 | 3 | 2023-03-10 | 2024-03-10 | Activo |
| 4 | 4 | 2 | 2023-02-15 | 2023-03-15 | Activo |
| 5 | 5 | 3 | 2023-03-10 | 2024-03-10 | Activo |
| 6 | 3 | 3 | 2023-03-10 | 2024-03-10 | Activo |

**Tabla de Pagos:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID\_Pago** | **ID\_Poliza** | **Fecha\_Pago** | **Total\_ Pago** | **Moneda** |
| 1 | 1 | 2023-01-05 | 500 | PEN |
| 2 | 2 | 2023-02-20 | 200 | USD |
| 3 | 3 | 2023-03-12 | 300 | PEN |
| 4 | 4 | 2023-01-05 | 200 | PEN |
| 5 | 5 | 2023-02-20 | 300 | USD |
| 6 | 6 | 2023-03-12 | 300 | PEN |

**Tabla de Pagos Detalle:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID\_Detalle\_Pago** | **ID\_Pago** | **Monto\_Pago\_Detalle** | **Metodo\_Pago** | **Moneda** |
| 1 | 1 | 280 | Tarjeta de Débito | PEN |
| 2 | 1 | 220 | Transferencia Bancaria | PEN |
| 3 | 2 | 110 | Tarjeta de Débito | USD |
| 4 | 2 | 90 | Tarjeta de Débito | USD |
| 5 | 3 | 300 | Tarjeta de Débito | PEN |
| 6 | 4 | 100 | Tarjeta de Débito | USD |
| 7 | 4 | 100 | Tarjeta de Débito | USD |
| 8 | 5 | 300 | Tarjeta de Débito | PEN |
| 9 | 6 | 300 | Tarjeta de Débito | PEN |

**Donde:**

**Tabla de Coberturas:**

Esta tabla almacena información sobre los diferentes tipos de coberturas que ofrece la compañía de seguros. Cada registro representaría una cobertura específica, como seguro de vida, seguro de salud, seguro de automóvil, entre otros. Los campos que contiene son:

|  |  |
| --- | --- |
| ID\_Cobertura: | Un identificador único para cada tipo de cobertura (clave primaria). |
| Nombre\_Cobertura: | El nombre descriptivo de la cobertura. |
| Descripcion: | Una breve descripción que detalla qué está cubierto por esa póliza. |

**Tabla de Seguros:**

En esta tabla se registran los detalles de los diferentes seguros disponibles para los clientes. Cada seguro estaría asociado a una cobertura específica de la tabla anterior. Los campos en esta tabla son:

|  |  |
| --- | --- |
| ID\_Seguro: | Un identificador único para cada seguro (clave primaria). |
| ID\_Cobertura: | Un identificador que establece la relación con la tabla de Coberturas (clave foránea). |
| Nombre\_Seguro: | El nombre o título del seguro. |
| Descripcion: | Información adicional que describe la póliza en detalle. |
| Precio: | El costo del seguro para el período de cobertura. |
| Moneda: | La moneda del costo del seguro para el período de cobertura (PEN=Moneda Peruana o USD=Dólar). |
| Duracion: | Indica la duración de la póliza (por ejemplo, anual, mensual, trimestral, semestral, etc.). |

**Tabla de Clientes:**

En esta tabla se almacenan los datos de los clientes que adquieren los seguros. Cada registro representa un cliente distinto. Los campos que contiene son:

|  |  |
| --- | --- |
| ID\_Cliente: | Un identificador único para cada cliente (clave primaria). |
| Nombre\_ Cliente: | El nombre del cliente. |
| Direccion: | La dirección postal del cliente. |
| Correo\_Electronico: | La dirección de correo electrónico del cliente. |
| Número\_Telefono: | El número de teléfono del cliente. |
| Fecha\_Nacimiento: | La fecha de nacimiento del cliente. |
| Sexo: | El sexo del cliente (M=Masculino o F=Femenino). |

**Tabla de Pólizas:**

Esta tabla se utiliza para registrar las pólizas individuales que se han vendido a los clientes. Los campos incluidos son:

|  |  |
| --- | --- |
| ID\_Poliza: | Un identificador único para cada póliza (clave primaria). |
| ID\_Cliente: | Un identificador que enlaza la póliza con un cliente específico de la tabla de Clientes (clave foránea). |
| ID\_Seguro: | Un identificador que enlaza la póliza con el tipo de seguro correspondiente de la tabla de Seguros (clave foránea). |
| Fecha\_Inicio: | La fecha en que la póliza comienza a ser válida. |
| Fecha\_Vencimiento: | La fecha en que la póliza expira o vence. |
| Estado\_Poliza: | Indica si la póliza está activo, cancelado, vencido o anulado. |

**Tabla de Pagos:**

En esta tabla se registran los pagos realizados por los clientes para mantener sus pólizas activas. Los campos son:

|  |  |
| --- | --- |
| ID\_Pago: | Es el identificador único (clave primaria) del pago. Se utiliza para identificar de manera exclusiva cada registro en la tabla. |
| ID\_Poliza: | Es una clave externa (foreign key) que hace referencia al identificador único de una póliza en la tabla "Polizas". Esto establece una relación entre el pago y la póliza a la que pertenece. |
| Fecha\_ Pago: | Es la fecha en la que se realizó el pago. Se almacena como un tipo de dato DATE. |
| Total\_Pago: | Es el monto total pagado en la transacción. Se almacena como un número decimal con una precisión de 10 dígitos y 2 decimales. |
| Moneda\_Pago: | Es la moneda del monto total pagado en la transacción. |

**Tabla de Detalle Pagos:**

|  |  |
| --- | --- |
| ID\_Detalle\_Pago: | Es el identificador único (clave primaria) del detalle de pago. Cada registro en esta tabla tiene un ID único para identificarlo de manera exclusiva. |
| ID\_Pago: | Es una clave externa (foreign key) que hace referencia al identificador único de un pago en la tabla "Pagos". Esto establece una relación entre el detalle de pago y el pago al que pertenece. |
| Monto\_Pago\_Detalle: | Representa el importe que se pagó nivel de detalle. La suma de todos los montos de pago detalle debe ser la suma total del total del pago. |
| Metodo\_Pago: | Es una cadena que indica el método utilizado para realizar el pago, por ejemplo, "Tarjeta de Crédito", "Transferencia Bancaria", "Tarjeta de Débito", etc. |
| Moneda\_Pago: | Es la moneda del monto pagado a nivel de detalle en la transacción. |

**Requerimientos:**

1. Se requiere crear un procedimiento almacenado que muestre en un cursor de salida la cantidad total de pagos y sus montos totales que se realizaron en un periodo de fechas determinadas. **Recibirá como datos de entrada la fecha de inicio y fecha de fin a buscar**. La estructura del cursor de salida debe ser la siguiente.



1. Se requiere crear un procedimiento almacenado que según la fecha actual y el pago de las pólizas actualice el estado de la póliza a Activo o Desactivada por Impago. El procedimiento no cuenta con variables de entrada, se debe validar a través de la fecha actual.

Caso 1: Si la póliza no presenta registros de pagos en el mes de la fecha actual se debe actualizar el estado de la póliza a ‘Desactivada por Impago’.

Caso 2: Si la fecha de vencimiento de la póliza es menor a la fecha actual se deberá actualizar el estado de la póliza a ‘Finalizada’.

1. Se requiere crear un procedimiento almacenado que genere un reporte de pólizas el cual reciba distintos filtros de búsqueda y que dependiendo de si el filtro tiene o no valor en la variable de entrada se use o no este filtro. (Usar querys dinámicos)

Filtros de búsqueda de pólizas:

* Por moneda de la póliza
* Por duración de la póliza
* Por nombre del cliente
* Por sexo del cliente
* Por el nombre del seguro
* Por estado de la póliza

Los datos a mostrar serán los siguientes:

* POLIZA
* CLIENTE
* SEXO CLIENTE
* FECHA INICIO DE POLIZA
* FECHA FIN DE POLIZA
* ESTADO DE POLIZA
* NOMBRE COBERTURA
* NOMBRE SEGURO
* MONEDA POLIZA
* DURACION POLIZA
* MONTO DE LA POLIZA
* CANTIDAD TOTAL DE PAGOS POR POLIZA
* MONTO TOTAL DE PAGOS POR POLIZA

**Notas:**

* Para realizar la prueba, el candidato deberá tener acceso a un servidor Oracle (local o remoto) con las tablas mencionadas (Coberturas, Seguros, Clientes, Pólizas, Pagos y Pagos Detalle) creadas previamente.
* Puede tomar como referencia el script adjunto a este documento, que se le ha proporcionado para la creación de las tablas e inserción de la información en la Base de Datos.
* Recordar que cada requerimiento de procedimiento almacenado debe ser creado tanto en Oracle como en PostgreSQL.

**Criterios de Evaluación:**

* Los procedimientos almacenados deben poder ser interpretados de manera sencilla y clara por parte del evaluador.
* De ser el caso hacer uso de funciones o métodos nativos de cada motor de base de datos (pivot, crosstab, etc.).
* Las consultas deben estar lo mejor optimizadas para el procesamiento masivo de datos.
* Uso de buenas prácticas de codificación.

Deberás proporcionar los scripts para cada motor de base de datos (Oracle y PostgreSQL) en forma ordenada, así como scripts que permitan testear cada procedimiento almacenado.

¡Buena Suerte!