1.

FECHA	DESCRIPCION	IMPORTE PESOS	IMPORTE DOLARES
21/11/2011	Extracción	-1000.00	
12/21/2011	Depósito		1000,00
01/10/2011	Débito por transferensia	500.00	

Los defectos detectados en la presente tabla son:

a. El desfase y falta de alineación entre las columnas. El nombre “DOLARES” perteneciente a la última columna se encuentra alineado bajo el título de la primera columna. El monto en dólares que debería aparecer enmarcado en la última comuna aparece entre las fechas. Y los importe en pesos, si bien están en la columna correspondiente, no se encuentran alineados.
b. Incongruencia en el formato de la fecha. La primera fecha se encuentra en formato dd/mm/aaaa, mientras que la segunda está en formato mm/dd/aaaa. En este caso deberíamos remitirnos a la documentación para asegurar el tipo de formato que se debería utilizar en la tabla.
c. El formato de la descripción no parece aceptar caracteres especiales. La descripción de la segunda fila no queda clara. Es probable que el texto original haya tenido un carácter especial (en este caso una tilde) que no fue admitido al momento de generarse la tabla.
d. Incongruencia en el formato importe. En los importes bajo la columna “IMPORTE PESOS” los dos decimales están separados por punto (ejemplo 500.00), mientras que el importe correspondiente a la columna “IMPORTE DOLARES” los decimales están separados por coma (ejemplo: 1000,00).
e. Error de ortografía. La palabra transferencia en la descripción de la tercera fila está escrita con “s”.

2. Los campos a utilizar para el reporte de defectos serían: ID, Título, Descripción, Resultado obtenido, Resultado esperado y Evidencia.

ID	TÍTULO	DESCRIPCIÓN	RESULTADO OBTENIDO	RESULTADO ESPERADO	EVIDENCIA
1	Error de alineación de la columna “importe en dolares”	Error de márgenes provoca que la palabra dólares y el monto en dólares cambie de columna, mezclados con el contenido de la columna “fecha”	La palabra “dólares” pertenecientes al título de la última columna al igual que el importe en dólares se encuentran alineados bajo la columna fecha	El título “importe dólares” y el monto en dólares deberían están alineados dentro de una misma columna	

2	Falta de alineación en la columna “importe en pesos”	Los importes correspondientes a la columna “importe en pesos” están corridos (desalineados) el uno respecto al otro	El primer importe (-1000) se encuentra más adelante en la columna que el segundo importe (500)	Ambos números deberían estar alineados bajo el título de la columna y comenzar bajo una misma línea	
3	Incongruencia en el formato de fecha	No coincide el formato de las fechas de las distintas columnas. Nota: Debería tomarse en cuenta la documentación para saber qué formato es el correcto	El formato de la segunda fila es mm/dd/aaaa	Todas las filas deberían tener el formato dd/mm/aaaa	
4	Incongruencia en el formato decimal	No coincide el formato con el que se separan los decimales para los montos en pesos (se utiliza punto) y para los montos en dólares (se utiliza coma)	Los decimales en el monto en dólares están separados por coma (ejemplo: 1000,00)	Todos los montos deberían estar separados por punto (ejemplo 1000.00)	
5	Caracteres especiales	Al no aceptar el uso de caracteres especiales en la descripción de la segunda fila, provoca que la palabra quede ilegible	La descripción de la segunda fila se lee: Depósito	Debería leerse: Depósito	
6	Error ortográfico	La descripción de la tercer fila tiene errores de ortografía	La palabra “transferensia” está escrita con “S”	La palabra transferencia debería estar escrita con “C”	

- En la columna de “EVIDENCIA” se debería adjuntar una captura de pantalla donde se resalta solamente el error especificado en ese caso. No se realiza en este caso ya que no hay suficiente espacio en la tabla de word para colocar una imagen que se vea clara.

3.

a. Pruebas funcionales:Este tipo de pruebas se enfocan en verificar cómo funciona el sistema. Es decir, que el sistema cumpla con los requisitos especificados para cumplir

con su función y que haga lo que debe.
Tomemos por ejemplo un login:
Un caso de prueba positivo sería ingresar en email válido en el campo de Email. Ingresar una contraseña válida para el mail ingresado en el campo de Password. Presionar el botón de Ingresar. Y que se redirija al usuario a la página principal, o sea, que el login sea exitoso.
Un caso de prueba negativo sería que al ingresar un usuario inválido con una contraseña válida y presionar el botón de "ingresar", el sistema no nos permita el ingreso y nos advierta del error.

b. Pruebas no funcionales: Se enfocan en cómo un sistema debe funcionar. Dentro de este grupo podemos encontrar las pruebas de performance, de accesibilidad y de seguridad.
Siguiendo con el ejemplo del Login. Una prueba no funcional sería verificar el tiempo de carga de la pantalla del login o cuanto tiempo tarda el sistema en dirigirnos a la página principal una vez ingresado.

c. Pruebas de carga y estrés: Este tipo de pruebas funcionales se enfocan en ver cómo reacciona el sistema frente a una determinada demanda. Los test de carga se centran en ver que el sistema se comporte adecuadamente bajo la cantidad estipulada de demanda, mientras que la pruebas de estrés verifican que sucede cuando se exceden los límites normales (o estipulados) para el sistema (por ejemplo ver si el sistema se puede recuperar luego de una demanda elevada o a partir de qué punto comienza a fallar)
Ejemplo: Supongamos que tenemos una sistema que corre por Cron cada 12 horas y tiene estimado recibir entre 50 y 100 records por corrida. Para la prueba de carga, se podrían enviar 100 records a la vez al sistema para ver que se procesen de manera adecuada, mientras que para la prueba de estrés se podrían enviar 200 archivos, para ver como se comporta el sistema.

4. Se deberían realizar 5 particiones de equivalencia para cubrir los casos. Deberíamos probar un número menor a cero para asegurarnos que el sistema no admite números negativos, otro entre cero y menos que dos para el segundo grupo, otro entre 2 pero menor a cinco, el cuarto entre cinco y menor a diez, y el último un número mayor o igual a 10.

5.

a. Verdadero. Ningún escenario contempla el flujo D → F
b. Verdadero. No se prueba el flujo D → F
c. Falso. Tanto el ingreso de datos D → E como sus dos posible egresos hacia F o G fueron testeados en las decisiones A y B
d. Falso. Los resultados de la decisión F hacia G o C están contemplados en los test B y C.

6. Para estimar el esfuerzo que requiere testear un requerimiento me baso en el análisis de alto nivel del mismo. Con este análisis busco comprender la complejidad del requerimiento, por ejemplo en el caso de una API, cuantos records habrá por llamado, que tipo de trigger se utiliza, si la solución requiere alguna transformación, si se utiliza algún tipo de autenticación y cual. En caso de que se manejen archivos, saber si estos pasan por encriptación o compresión y que cantidad se prevé por llamado. Así como también la complejidad que tendrá el mapeo.

7. Se adjunta colección de Postman
Lógica:
Se crean las variables de colección para LOCATION, bonos y otra (index) para recorrer el array de bonos.
En el pre-request se obtiene el array y el indice y eso se vuelca a una variable (bono sobre el que estamos iterando). Luego se hace una función para iterar sobre el array y que al finalizar vuelva a dejar el índice en cero.
En el post-request se evalúan los siguientes casos:

- Que la key “shortName” sea igual al nombre del bono que viene en el endpoint.
- Que “name”, “debtType” y “self” no tengan un valor igual a null.
- Que las URLs de los otros componentes del objeto “link” sean iguales a la URL de self y agreguen el nombre de la key al final.