Functionality:
At the beginning, there is no account, so every transaction on an invalid id will be an error.

```
xml content length:95
xml content:<transactions id="1234567890">
    <order sym="gu2" amount="100" limit="10"/>
</transactions>

Message sent: 99 bytes
Message received: 132 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <error sym="gu2" amount="100" limit="10">Account not exists</error>
</results>
```

Step1:
Create 3 accounts
1234567890, balance 10000, gu1 100, gu2 100
1234567891, balance 10000, gu1 100, gu2 100
1234567892, balance 200, gu1 0, gu2 0
Cover corner cases like symbol assigned to an invalid account, balance <0, symbol amount <0, account creation duplicated.

```
test_create1.xml
xml content length:605
xml content:<create>
    <account id="1234567890" balance="10000"/>
    <account id="1234567891" balance="10000"/>
    <symbol sym="gu1">
        <account id="1234567890">100</account>
        <account id="1234567891">-100</account>
        <account id="1234567892">100</account>
    </symbol>
    <symbol sym="gu2">
        <account id="1234567890">100</account>
        <account id="1234567891">100</account>
        <account id="1234567891">100</account>
    </symbol>
    <account id="1234567892" balance="-100"/>
    <account id="1234567892" balance="200"/>
    <account id="1234567892" balance="0"/>
</create>

Message sent: 610 bytes
Message received: 574 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <created id="1234567890"/>
    <created id="1234567891"/>
    <created sym="gu1" id="1234567890"/>
    <error sym="gu1" id="1234567891">Symbol amount less than 0 </error>
    <error sym="gu1" id="1234567892">Account not exists</error>
    <created sym="gu2" id="1234567890"/>
    <created sym="gu2" id="1234567891"/>
    <created sym="gu2" id="1234567891"/>
    <error id="1234567892">Account balance less than 0</error>
    <created id="1234567892"/>
    <error id="1234567892">Account already exists</error>
</results>
```

Step2: Create 3 accounts
1234567893, balance 10000, gu1 1000, gu2 0
1234567894, balance 10000, gu1 0, gu2 500
1234567895, balance 10000, gu1 0, gu2 500
Corner cases including symbol name invalid(-+gu3)

```
xml content length:509
xml content:<create>
    <account id="1234567893" balance="10000"/>
    <account id="1234567894" balance="10000"/>
    <account id="1234567895" balance="10000"/>
    <symbol sym="gu1">
        <account id="1234567893">1000</account>
    </symbol>
    <symbol sym="gu2">
        <account id="1234567894">500</account>
        <account id="1234567895">500</account>
    </symbol>
    <symbol sym="-+gu">
        <account id="1234567894">500</account>
        <account id="1234567895">500</account>
    </symbol>
</create>

Message sent: 514 bytes
Message received: 408 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <created id="1234567893"/>
    <created id="1234567894"/>
    <created id="1234567895"/>
    <created sym="gu1" id="1234567893"/>
    <created sym="gu2" id="1234567894"/>
    <created sym="gu2" id="1234567895"/>
    <error sym="-+gu" id="1234567894">Symbol name invalid</error>
    <error sym="-+gu" id="1234567895">Symbol name invalid</error>
</results>
```

Step3: Create 3 accounts
1234567896, balance 10000, gu1 0, gu2 300
1234567897, balance 10000, gu1 0, gu2 300
1234567898, balance 10000, gu1 1000, gu2 0

```
xml content length:377
xml content:<create>
    <account id="1234567896" balance="10000"/>
    <account id="1234567897" balance="10000"/>
    <account id="1234567898" balance="10000"/>
    <symbol sym="gu1">
        <account id="1234567898">1000</account>
    </symbol>
    <symbol sym="gu2">
        <account id="1234567897">300</account>
        <account id="1234567896">300</account>
    </symbol>
</create>

Message sent: 382 bytes
Message received: 276 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <created id="1234567896"/>
    <created id="1234567897"/>
    <created id="1234567898"/>
    <created sym="gu1" id="1234567898"/>
    <created sym="gu2" id="1234567897"/>
    <created sym="gu2" id="1234567896"/>
</results>
```

All valid

Start transactions:
Step1:
Open order for 1234567890, transaction id 1-> buy gu1 100 $8
Open order for 1234567890, transaction id 2-> buy gu1 300 $7

```
xml content:<transactions id="1234567890">
    <order sym="gu1" amount="100" limit="8"/>
    <order sym="gu1" amount="300" limit="7"/>
    <query id="1"/>
</transactions>

Message sent: 165 bytes
Message received: 231 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <opened sym="gu1" amount="100" limit="8" id="1"/>
    <opened sym="gu1" amount="300" limit="7" id="2"/>
    <status id="1">
        <open shares="100"/>
    </status>
</results>
```

Step2:
All for corner cases:
gu3 not exists, query other account transaction, cancel other account transaction.
open transaction on gu3 for account 1234567891(gu3 not exists)
query transaction id 1 which is on account 1234567890
query transaction id 100 (invalid)
cancel transaction id 100 and 1

```
xml content:<transactions id="1234567891">
    <order sym="gu3" amount="100" limit="10"/>
    <order sym="gu3" amount="-100" limit="10"/>
    <query id="1"/>
    <query id="100"/>
    <cancel id="100"/>
    <cancel id="1"/>
</transactions>

Message sent: 234 bytes
Message received: 464 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <error sym="gu3" amount="100" limit="10">Symbol not exists</error>
    <error sym="gu3" amount="-100" limit="10">Symbol not exists</error>
    <error id="1">Query a transaction not belonging to this account</error>
    <error id="100">Transaction id not exists</error>
    <error id="100">Transaction id not exists</error>
    <error id="1">Cancel a transaction not belonging to this account</error>
</results>
```

Step3:
All for corner cases: price<0, insufficient share, and balance.

```
test_transaction3.xml
xml content length:239
xml content:<transactions id="1234567892">
    <order sym="gu2" amount="100" limit="10"/>
    <order sym="gu2" amount="100" limit="-10"/>
    <order sym="gu1" amount="-100" limit="10"/>
    <order sym="gu2" amount="-100" limit="-8"/>
</transactions>

Message sent: 244 bytes
Message received: 395 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <error sym="gu2" amount="100" limit="10">Insufficient Balance for the order</error>
    <error sym="gu2" amount="100" limit="-10">price less than 0 invalid</error>
    <error sym="gu1" amount="-100" limit="10">Insufficient share for the order</error>
    <error sym="gu2" amount="-100" limit="-8">price less than 0 invalid</error>
</results>
```

Step4:
Open order for account 1234567893,
order id 3 = sell 500 gu1 price 11, order id 4 = sell 100 gu1 price 12

```
test_transaction4.xml
xml content length:144
xml content:<transactions id="1234567893">
    <order sym="gu1" amount="-500" limit="11"/>
    <order sym="gu1" amount="-100" limit="12"/>
</transactions>

Message sent: 149 bytes
Message received: 172 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <opened sym="gu1" amount="-500" limit="11" id="3"/>
    <opened sym="gu1" amount="-100" limit="12" id="4"/>
</results>
```

Step5:
Open order for account 1234567894,
order id 5 = sell 200 gu1 price 15, order id 6 = sell 200 gu1 price 20

```
test_transaction5.xml
xml content length:184
xml content:<transactions id="1234567894">
    <order sym="gu2" amount="-200" limit="15"/>
    <order sym="gu2" amount="-200" limit="20"/>
    <query id="6"/>
    <query id="7"/>
</transactions>

Message sent: 189 bytes
Message received: 288 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <opened sym="gu2" amount="-200" limit="15" id="5"/>
    <opened sym="gu2" amount="-200" limit="20" id="6"/>
    <status id="6">
        <open shares="-200"/>
    </status>
    <error id="7">Transaction id not exists</error>
</results>
```

Step6:
Open order for account 1234567895,
order id 7 = sell 200 gu1 price 15, order id 8 = sell 200 gu1 price 20

```
test_transaction6.xml
xml content length:144
xml content:<transactions id="1234567895">
    <order sym="gu2" amount="-200" limit="15"/>
    <order sym="gu2" amount="-200" limit="20"/>
</transactions>

Message sent: 149 bytes
Message received: 172 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <opened sym="gu2" amount="-200" limit="15" id="7"/>
    <opened sym="gu2" amount="-200" limit="20" id="8"/>
</results>
```

Step7:
Open order for account 1234567896,
order id 9 = buy 100 gu1 price 9, order id 10 = sell 100 gu1 price 10

```
test_transaction7.xml
xml content length:141
xml content:<transactions id="1234567896">
    <order sym="gu1" amount="100" limit="9"/>
    <order sym="gu1" amount="100" limit="10"/>
</transactions>

Message sent: 146 bytes
Message received: 170 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <opened sym="gu1" amount="100" limit="9" id="9"/>
    <opened sym="gu1" amount="100" limit="10" id="10"/>
</results>
```

Step8:
Open order for account 1234567897,
order id 11 = buy 100 gu2 price 9

```
test_transaction8.xml
xml content length:95
xml content:<transactions id="1234567897">
    <order sym="gu2" amount="100" limit="15"/>
</transactions>

Message sent: 99 bytes
Message received: 116 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <opened sym="gu2" amount="100" limit="15" id="11"/>
</results>
```

Step9:
Open order for account 1234567898,
order id 12 = sell 1000 gu1 price 6

```
test_transaction9.xml
xml content length:96
xml content:<transactions id="1234567898">
    <order sym="gu1" amount="-1000" limit="6"/>
</transactions>

Message sent: 100 bytes
Message received: 117 bytes
Message received: <?xml version="1.0" encoding="UTF-8"?>
<results>
    <opened sym="gu1" amount="-1000" limit="6" id="12"/>
</results>
```

In the end we can go to the database and see the result

```
trades=# SELECT * FROM account;
 account_id | balance
------------+---------
 1234567891 |   10000
 1234567892 |     200
 1234567893 |   10000
 1234567895 |   10000
 1234567890 |    7100
 1234567896 |    8100
 1234567897 |    8500
 1234567894 |   11500
 1234567898 |   14800
(9 rows)
```

```
trades=# SELECT * FROM position;
 account_id | symbol_name | amount
------------+-------------+--------
 1234567890 | gu2         |    100
 1234567891 | gu2         |    200
 1234567896 | gu2         |    300
 1234567893 | gu1         |    400
 1234567894 | gu2         |    100
 1234567895 | gu2         |    100
 1234567897 | gu2         |    400
 1234567898 | gu1         |      0
 1234567890 | gu1         |    500
(9 rows)
```

```
trades=# SELECT * FROM transaction;
 transaction_id |    time    | account_id |  status  | symbol_name | price | amount
----------------+------------+------------+----------+-------------+-------+--------
              3 | 1680834493 | 1234567893 | OPEN     | gu1         |    11 |   -500
              4 | 1680834493 | 1234567893 | OPEN     | gu1         |    12 |   -100
              6 | 1680834668 | 1234567894 | OPEN     | gu2         |    20 |   -200
              7 | 1680834785 | 1234567895 | OPEN     | gu2         |    15 |   -200
              8 | 1680834785 | 1234567895 | OPEN     | gu2         |    20 |   -200
              5 | 1680834668 | 1234567894 | OPEN     | gu2         |    15 |   -100
             11 | 1680834982 | 1234567897 | EXECUTED | gu2         |    15 |      0
             10 | 1680834863 | 1234567896 | EXECUTED | gu1         |    10 |      0
              9 | 1680834863 | 1234567896 | EXECUTED | gu1         |     9 |      0
              1 | 1680834111 | 1234567890 | EXECUTED | gu1         |     8 |      0
             12 | 1680835363 | 1234567898 | OPEN     | gu1         |     6 |   -400
              2 | 1680834111 | 1234567890 | EXECUTED | gu1         |     7 |      0
(12 rows)
```

Correct result

Overall:

ID	balance	gu1	gu2	orderid price amount
1234567890	10000	100	100	1($7)300 2($8)100 buy gu1
1234567891	10000	0	100	
1234567892	200	0	100	
1234567893	10000	1000	0	3($11)-500 4($12)-100 sell gu1
1234567894	10000	0	500	5($15)-200 6($20)-200 sell gu2
1234567895	10000	0	500	7($15)-200 8($20)-200 sell gu2
1234567896	10000	300	0	9 ($9)100 10($11)100 buy gu1
1234567897	10000	300	0	11($15)100 buy gu2
1234567898	10000	1000	0	12(6)-1000 sell gu1

For gu1

	4 $12 -100
10 $11 100	3 $11 -500
9 $9 100	
2 $8 100	
1 $7 300	
	12 $6 -1000

Thus, order 11 will buy 100 gu1 from order 3, with price $11.
And order 13 will sell 1000 gu1 for order 10, 2, 1 with price $9, $8, $7

For gu2

	6 $20 -200 8 $20-200
11 $15 100	5 $15 -200 7 $15 -200

Order 12 will buy 100 gu2 from order6 with price $15

Result:
the account 1234567890 will have 500gu1, 100 gu2 and 7100 balance in the end
the account 1234567891 will have 1000gu1, 0 gu2 and 10000 balance in the end
the account 1234567892 will have 0 gu1, 100 gu2 and 200 balance in the end
the account 1234567893 will have 900 gu1, 0 gu2 and 11000 balance in the end
the account 1234567894 will have 0 gu1, 400 gu2 and 11500 balance in the end
the account 1234567895 will have 0 gu1, 500 gu2 and 10000 balance in the end
the account 1234567896 will have 500 gu1, 0 gu2 and 8000 balance in the end
the account 1234567897 will have 300 gu1, 100 gu2 and 7500 balance in the end
the account 1234567898 will have 500 gu1, 0 gu2 and 13800 balance in the end

so, the serialized test is 100% correct.