



# Опыт использования технологии KDB+/Q в Дойче Банке

*Passion to Perform*

Андрей Бабанин  
25.10.2013

# Возрастание объемов биржевых данных



- *За последние 10 лет объемы биржевых данных возросли на **несколько порядков***



# Предпосылки



*С годами становилась очевидной неспособность традиционных реляционных баз данных обрабатывать такие гигантские объемы данных в реальном времени:*

- Классические реляционные СУБД не способны эффективно принимать и сохранять десятки и даже сотни тысяч записей в секунду*
- Реляционные СУБД не используют специальные подходы к обработке упорядоченных по времени данных*
- Традиционные языки программирования также либо неэффективны, либо неудобны при обработке гигантских массивов данных*

# Технология KDB+



- Основная разработка и продукт Kx Systems компании – технология **KDB+**
- KDB+ предоставляет **единый подход для обработки данных в памяти и на диске**
- В KDB+ встроен мощный интерпретируемый **язык Q**
- За годы развития и продвижения технологии клиентами Kx Systems стали **все крупнейшие инвестиционные банки и компании мира**

[www.kx.com](http://www.kx.com)

The screenshot shows the Kx Systems website. On the left is the Kx logo, with the tagline "The leading provider of Big time-series Data and analytics". To the right is a list of clients, each followed by a plus sign (+).

Goldman Sachs +	Bank of America +	ConvergEx +
Morgan Stanley +	Oppenheimer Capital Markets +	Connor, Clark & Lunn Financial Group +
Merrill Lynch +	UniCredit Bank AG +	RBC Capital Markets +
J.P. Morgan +	Zurich Financial Group +	GSA Capital +
NYSE/Euronext +	Rand Merchant Bank +	Total Gas & Power UK +
Deutsche Bank +	DekaBank +	US Army +
Commerzbank AG +		

# Производительность



- *KDB+* создана специально для 64-разрядной архитектуры
- **Встроенная многопоточность**
- Программный код *KDB+* **крайне компактен** и оптимизирован
- *KDB+* использует **расширенные наборы инструкций** процессоров Intel
- Тесты показывают **многократное превышение в производительности**

11th June 2013



## Kx's kdb+ claims up to 8x faster than any previous STAC M3 benchmarks

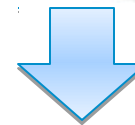
The results of the STAC (Securities Technology Analysis Center) M3™ vendor-independent market data benchmark suite demonstrate performance improvements in v3.1, with results said to be up to 8x faster than any previous STAC M3 benchmarks.

"Kx's financial services customers require continuous performance increases to keep pace with the market. The improvements Kx has implemented with kdb+ enhance both scaling with cores and concurrency," said Garry Thall, Intel Director of Financial Services, North America. He added: "Instruction Set Parallelism delivers improved Floating Point performance through enabling AVX instructions providing critical performance gains for Kx Customers."

# Хранение данных



- **Встроенная кластеризация хранилища данных**
- **KDB+ является нереляционной колонко-ориентированной базой данных**
- **KDB+ объединяет алгоритмы обработки с базой данных в памяти и на диске**



**Единый алгоритмический подход к обработке данных независимо от способа их хранения**

# Обработка данных



- **Механизмы встроенной кластеризации**
- **KDB+ обеспечивает оптимизацию сложных insert и update запросов**
- *Типы данных, описывающие время – от месяца до наносекунд, являются встроенными в KDB+*
- **В KDB+ встроенна динамическая индексация строковых данных и возможность использования собственных механизмов индексации**

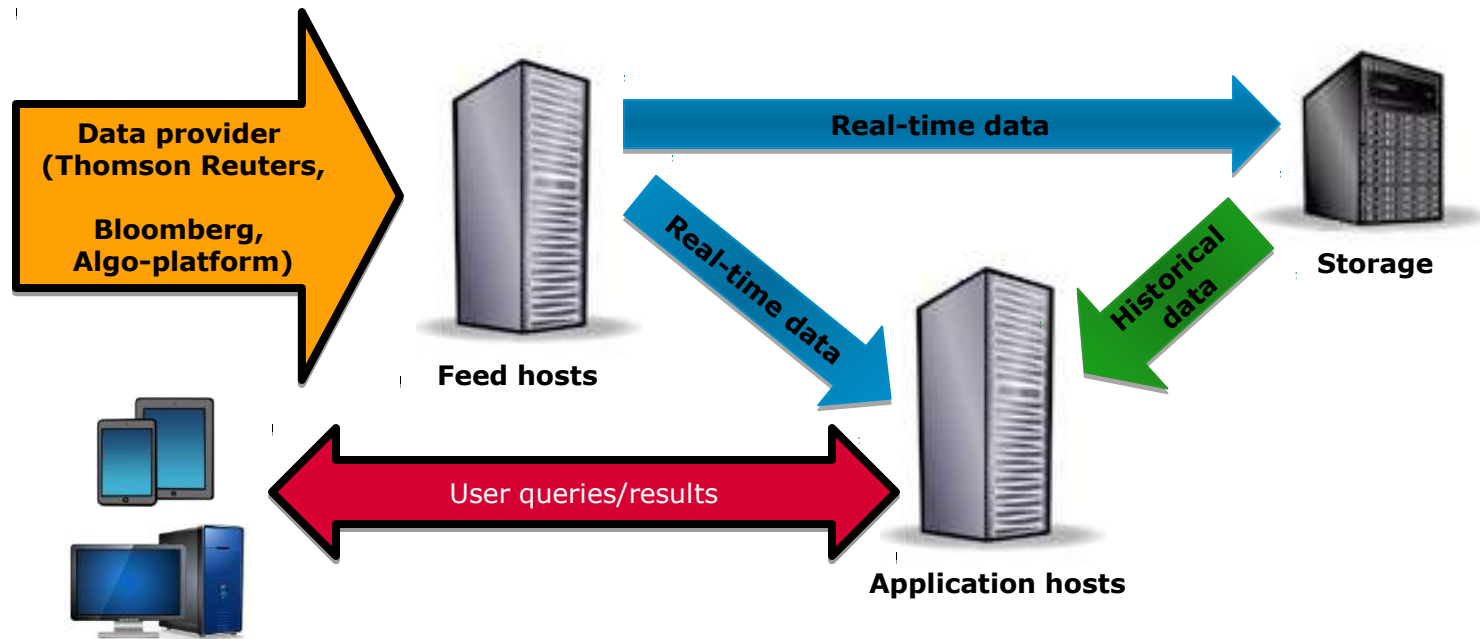
Особо стоит отметить наличие **двух видов строковых данных** в KDB+:

- *быстрые строки, который автоматически индексируются, и в операциях над ними используются только целочисленные индексы;*
- *обычный список одиночных символов; операции с ним намного затратнее по времени.*

# Передача данных



- *KDB+ рассчитана на создание конвейеров обработки*
- *Унифицированный файловый и сетевой ввод-вывод*
- *KDB+ поддерживает HTTP и WebSockets*





# Программные интерфейсы и расширения



- **KDB+** позволяет непосредственно загружать C код  
*имеет набор необходимых для разработки расширений заголовочных файлов*
- Имеет библиотеки для стыковки с приложения  
**C/C++, Java, .Net, R, Matlab, Perl, Python**
- KDB+ включает базовый инструментарий для подключения к провайдерам  
**Thomson Reuters** и **Bloomberg**
- Разработаны библиотеки для поддержки сетевых протоколов  
**Tibco, LBM, MAMA/MAMDA**
- Создан репозиторий всевозможных расширений языка на специальном ресурсе  
**code.kx.com**
- Для подключения к KDB+ серверам существует большое количество **сторонних оконных приложений с возможностями среды разработки и отладки**

# Встроенный язык Q



- Построен в первую очередь на **теории множеств**
- **Относится к классу немногословных языков**  
с односимвольными операциями и функциями
- **Является функциональным языком программирования**  
использует лямбда-исчисление
- Также **содержит процедурные элементы и реплику SQL**
- Поддерживает области видимости
- Использует **автоматическую типизацию**
- Интерпретатор поддерживает «уборку мусора»

```
rtcache: {[  
  system getenv[ `CURRENT ], "/bin/pjcheck.sh ", string[.z.D];  
  rr: ("SSS"; ",") 0: `$/:/tmp/pb1.csv";  
  rr[2]: {"T"$(" " vs string[x])[1]}each rr[2];  
  :`end_time xdesc update i from (flip `topid`sym`end_time!rr);  
  };  
  ...  
q: ([ host:4#enlist `;check:`FX_UPD_buf`LD_UPD_buf`FX_done_list`LD_done_list);  
  ...  
{.r[x; `vwap]: ((x+1)#.r[ `vol]) wavg ((x+1)#.r[ `vwap_price]);} each til count .r;
```

# Сильные стороны Q

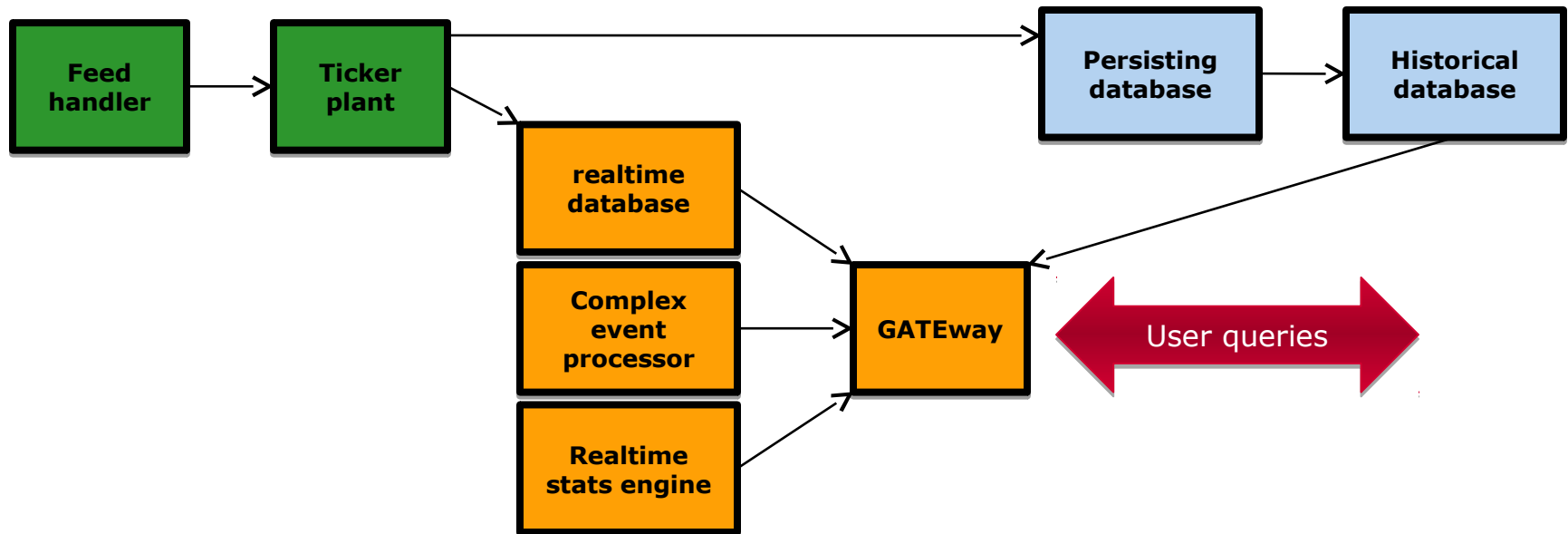


- *Q непосредственно работает с большими данными*
- *Списки, словари и таблицы являются базовыми типами данных*
- *Типы времени встроены в сам язык*
- *Для таблиц создан набор специальных атрибутов*
- *Q является интерпретируемым*
- *Хорошо интегрирован в среду Unix*
- *Время разработки меньше, чем на других языках*

# Основной сценарий использования KDB+ в Дойче Банке



- *KDB+ является базовой технологией для **глобального сбора и хранения биржевых и торговых данных** в Дойче Банке*
- *На основе KDB+ создано централизованное хранилище биржевых данных, которое одновременно используется и **для расчетов в реальном времени, и для исторического анализа***
- *Логика практически всех процессов **на 100% написана на языке Q***

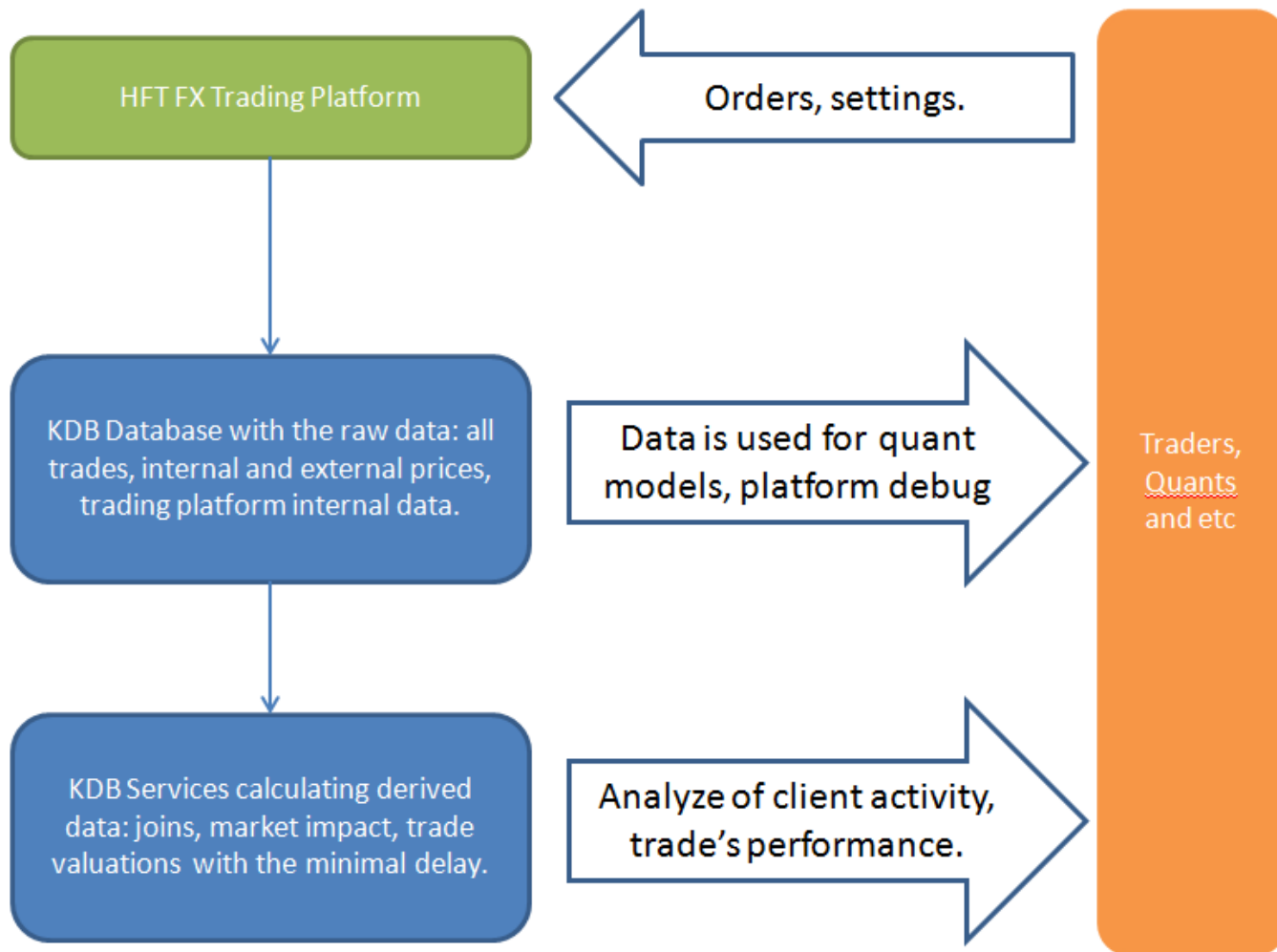


# Сбор и использование биржевых данных в цифрах

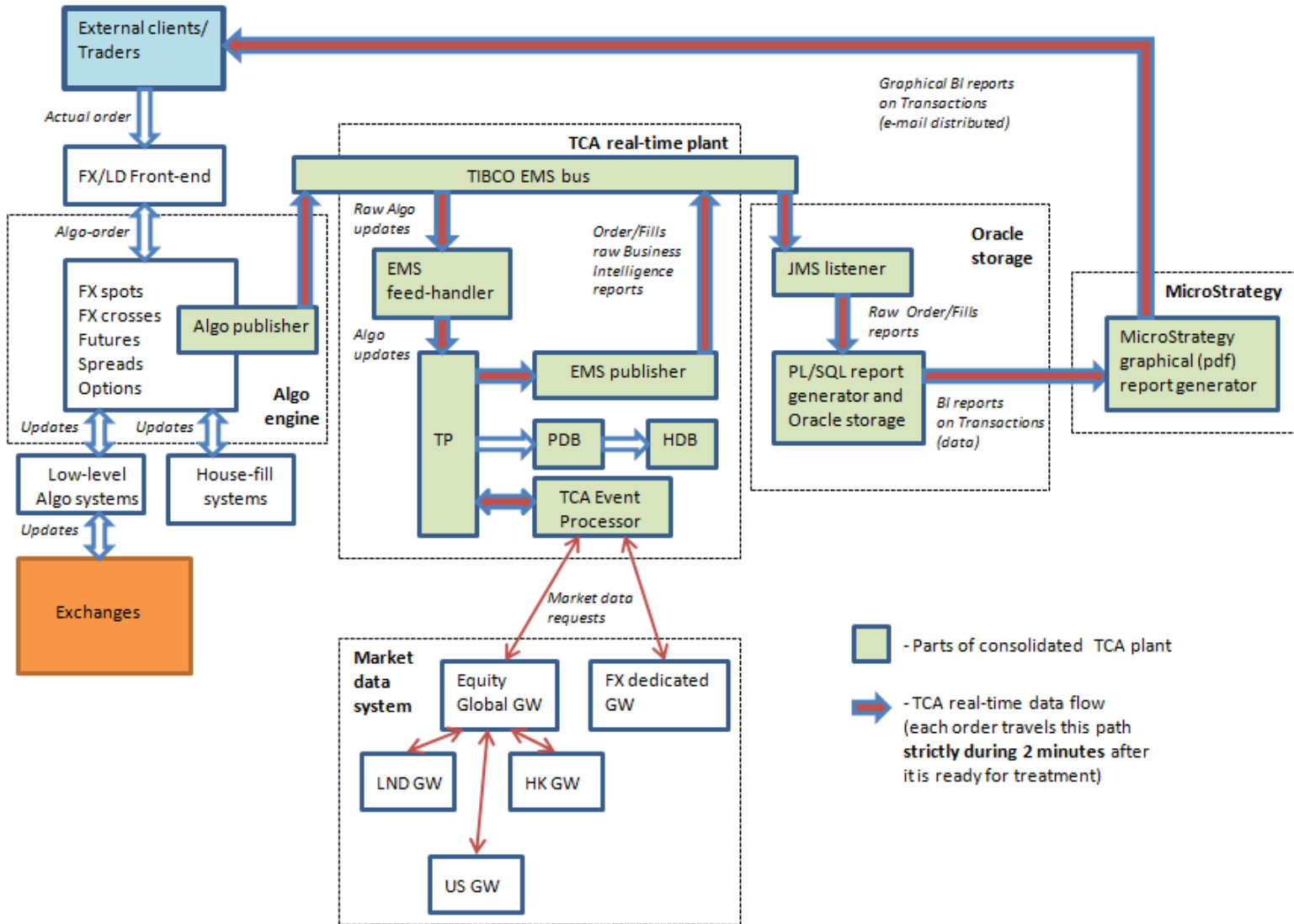


- **30 млрд. записей в день**
- **2,5 млн. активных подписок**
- **2 петабайта дискового пространства**
- **3000 активных процессов работающих в режиме 24x7**
- **Более 200 активных серверов**
- **Более 70 млн. внутренних запросов в день**
- **Несколько тысяч внутренних пользователей**
- **В системе создано несколько контуров, в том числе **активный резервный, эквивалентный основному, и контур тестирования****
- **Создано **несколько систем-спутников**, для обслуживания алгоритмической торговли**

# Использование KDB+ в системе сверхбыстрой торговли



# KDB+ в системе оценки затрат по транзакциям в реальном времени



# Отладчик для языка Q



Firefox - http://localhost:5566/debug.html

localhost:5566/debug.html

Step Into Step Over Step Out One Instr Continue  :check to enable native execution 9998 :number of exceptions to handle via the catch blocks

```
[[[arg] if[arg=50; "break"]; arg*arg][x]+1} each til 100} 1
```

Exec Run Prepare Show Fn Show Bps Clear Bps Refresh Help About

Current status: **exception**, Current function: anonymous, see the body below

```
.d.st[7] : "break"
.d.st[6] : Fn call, line: 0, fn: {[arg] if[arg=50; "break"]; arg*arg}[x]+1}
.d.st[5] : 1
.d.st[4] : Fn call, line: 14, fn: .d.each
.d.st[3] : 49
.d.st[2] : Fn call, line: 0, fn: {[arg] if[arg=50; "break"]; arg*arg}[x]+1} each
.d.st[1] : Fn call, line: 0, fn: {[arg] if[arg=50; "break"]; arg*arg}[x]+1}
.d.st[0] : Fn call, line: 0, fn: {"Entry point"}
```

```
.d.s["arg"] : 50
```

```
if[arg=50;
  "break";
  ::];
: arg * arg
```

User exception: break



# Среда разработки QPad



The screenshot shows the QPad development environment with a SQL script editor and an output table. The script editor contains the following SQL code:

```
save `:fo1009.xml`

select

last date

update

select i from @!
.ac1.users
where uname=`mouacam`

update utype=`root`,gname=`mariner` from @!.ac1.users where uname=`mouacam`

{@!.ac1.users}[56]
/a
-100#select from clientalgoorder

.ac1.users
:!(@!.ac1.users),('uname`fname`lname`email`hnames`utype`gname`white`black`perms')!(`mouacam2`;'Cameron`;'Mouat`;'');`root`;'mariner';(`symbol$()`)!`symbol
$()):( `symbol$()`)!`symbol$();(enlist `servertypes`)enlist `gw`cs));

.cashfxep.reprocess[ `clientalgoorder`; `FMG.UAT.C.01.P.1310021520000002585]

select from (select from .cashfxep.buffer[ `clientalgoorder` ] where id=`FMG.UAT.C.01.P.1310021520000002585` ) where not id in (exec distinct parentid from
(select from .cashfxep.buffer[ `clientalgoorder` ] where id=`FMG.UAT.C.01.P.1310021520000002585`),state in `2`3`4,executedquantity>0 / .cashfxep.buffer
```

The output table displays the following data:

date	time	seqnum	src	clientorderid	srcid	rootclientalgoorderid	level	book	side	sym	ccy	insttype	typ	rate	timeinforce
2013.10.11	2013.10.11D12:17:45.165000000	433835038177394893	FMG.UAT.C.01.P	338	884	883	2	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0
2013.10.11	2013.10.11D12:17:45.165000000	433835038177394894	FMG.UAT.C.01.P	338	885	1	2	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0
2013.10.11	2013.10.11D12:17:45.166000000	433835038177394895	FMG.UAT.C.01.P	3754	21026	885	2	ACEFXLN	2	NZDUSD	NZD	SPT	2	0.8336	0
2013.10.11	2013.10.11D12:17:45.166000000	433835038177394896	FMG.UAT.C.01.P	776	883	1	1	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0
2013.10.11	2013.10.11D12:17:45.166000000	433835038177394897	FMG.UAT.C.01.P	338	884	883	2	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0
2013.10.11	2013.10.11D12:17:47.031000000	433835038177394898	FMG.UAT.C.01.P	3755	21027	885	2	ACEFXLN	2	NZDUSD	NZD	SPT	2	0.8336	0
2013.10.11	2013.10.11D12:17:47.113000000	433835038177394899	FMG.UAT.C.01.P	338	885	1	1	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0
2013.10.11	2013.10.11D12:17:47.113000000	433835038177394900	FMG.UAT.C.01.P	3755	21027	885	2	ACEFXLN	2	NZDUSD	NZD	SPT	2	0.8336	0
2013.10.11	2013.10.11D12:17:47.113000000	433835038177394901	FMG.UAT.C.01.P	776	883	1	1	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0
2013.10.11	2013.10.11D12:17:47.113000000	433835038177394902	FMG.UAT.C.01.P	338	884	883	2	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0
2013.10.11	2013.10.11D12:17:47.114000000	433835038177394903	FMG.UAT.C.01.P	338	885	1	1	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0
2013.10.11	2013.10.11D12:17:47.113000000	433835038177394904	FMG.UAT.C.01.P	3755	21027	885	2	ACEFXLN	2	NZDUSD	NZD	SPT	2	0.8336	0
2013.10.11	2013.10.11D12:17:47.114000000	433835038177394905	FMG.UAT.C.01.P	776	883	1	1	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0
2013.10.11	2013.10.11D12:17:47.114000000	433835038177394906	FMG.UAT.C.01.P	338	884	883	2	ACEFXLN	2	NZDUSD	NZD	SPT	1	0	0



Спасибо!