○MQL5；翻訳「オーダー等情報入手の手順について（その１）」翻訳のみ実施　2012.10.28

・アメンボです、

MQL5 は MQL4 に比べて、「オーダーを出すにも、情報を入手するにも、
何て手間が掛かるのだろうか！」と、感じる諸兄が殆どでしょう。（略確信）
「手間と利益を秤にかけて」みたいのですが、現状では MQL5 の理解不足で
まだその段階には達していません。

・本稿では、MQL5 における「情報入手」の仕組みを翻訳・解説します。
（仕組みの一部分だけかもしれません。なにせ、全体が未だ見通せていないので！）

どうも MQL5 の概念（システム構成）を、理解できていないところが多々あるのですが、
現状のアメンボ理解として報告することにしました。（内容に間違いがあるかも！？）
（諸兄には、是非とも「概念理解」を修正・深化・発展させてください）

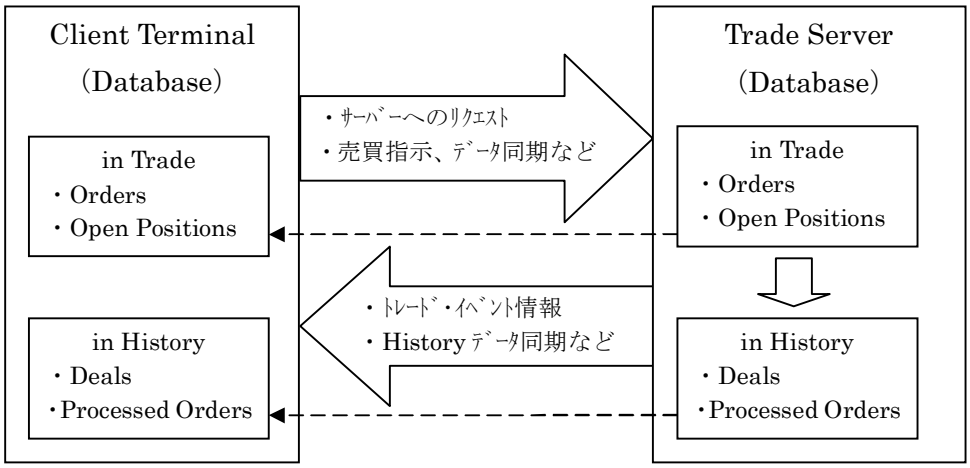
注意；　・本資料は、まだＭＴ５での動作・検証を行っていません、
・本編は近々の検証用資料として、英文資料を意訳しながら纏めたもの（メモ）です。
訳した資料がある程度たまったところで、ＭＴ５をダウンロードして確認して
いくつもりです。・・・すいません、まだＭＴ５は使ったことが無いのです！
（実機で未検証の内容ですので、誤訳があるかもしれません）
・以上の状況を理解されたうえで、本稿内容を参照ください。
○本稿を「(その１)」としたのは、基本内容のみを記述したので、別の機会に応用や実施例等を
報告しようと考えているからです。

目次：

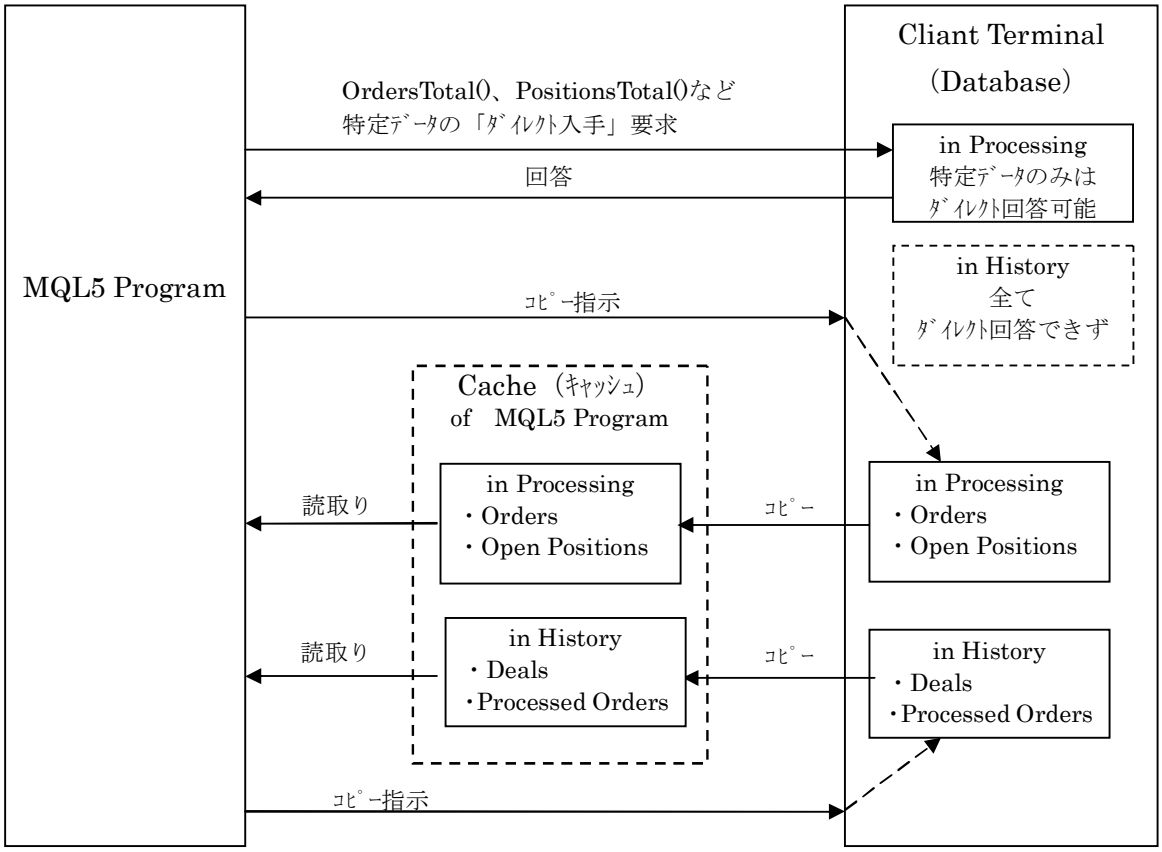
１．基礎知識の再確認

（1）ターミナルとサーバーと、そして「キャッシュ」

１）関係図１；　ターミナルはトレード・サーバーとデータの同期を実施



２）関係図２；キャッシュ（cache）の位置とその機能



※キャッシュはターミナルと MQL5 プログラムの中間に介在して、データの一時的「格納庫」の役割を果たす。

※MQL4 では、「in Processing」にしろ「in History」にしろ、ダイレクト（直接）にターミナル（Database）から情報を入手できた、が、

一方、MQL5 では、上図（関係図２）の関係にある、すなわち

・「現在進行形」のデータでも、MQL5 からダイレクトに得られるものは、限定される。
「ヒストリー」データでは、ダイレクトに得られるものは無い。
・詳細なデータは、全て「キャッシュ」経由で入手する、この場合、MQL5 側の処理は
①「履歴期間指定」→②「キャッシュへのコピー指示」→③「キャッシュから読取り」
の３段階となる。

２．情報を入手するための手順（概要）

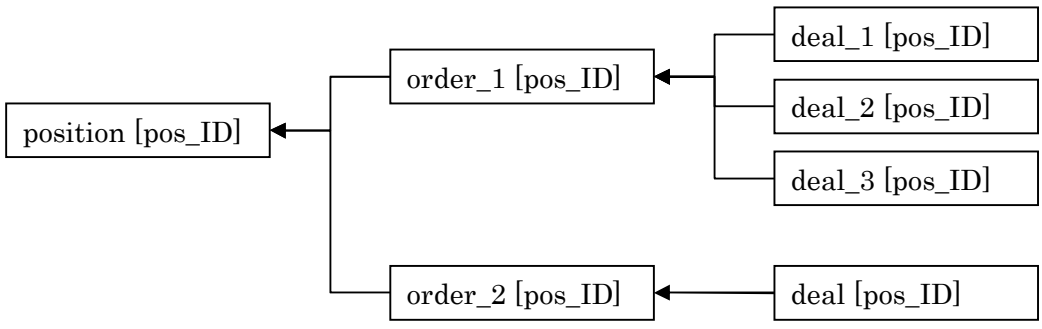
※MQL5 では、キャッシュを経由する情報（関数）と経由しない情報（関数）がある

（１）MQL4 と MQL5 の重要な違い・・・詳細検討に入る前に特徴を概観する

A．基本概念の違い

	order	deal	position
MQL4	・現在ｱｸﾃｨﾌﾞなｵｰﾀﾞｰと、決済済み（履歴）のｵｰﾀﾞの区別は無く、両者ともにｵｰﾀﾞｰと呼ぶ。	概念が無い	・一般的な意味での概念は在るが、MQL4 言語中には「用語が無い」
MQL5	An order is a request to conduct a transaction. ・ｵｰﾀﾞｰとは、処理実施の要求。	A deal is the reflection of the fact of a trade operation execution based on an order that contains a trade request. ・ﾃﾞｨｰﾙとは、ｵｰﾀﾞｰに基づくﾄﾚｰﾄﾞを実施した結果（反映）。	a position is a result of one or more deals. ・ﾎﾟｼﾞｼｮﾝは、1 つ又は複数のﾃﾞｨｰﾙで構成される。

※「order、deal、position」の関係・・理解してない部分が多く、原本（英語）も載せた。
・一つの position には、その構成要素である「order と deal」が存在し、それぞれは「pos_ID（position_ID）；position identifier」によって纏められ（束ねられ）ている。
つまり、「order と deal」には、それぞれが属する「position_ID」タグが付いている。



（２）MQL5 における「オーダー・データ等」の存在場所（形式）

	アクティブ・データ	ヒストリカル・データ	備　考
order	○	○	・ポジションの「オープン」用と「手仕舞い」用の２分類あり
deal	—	○	・個別の約定に分類なし
position	○	—	・開いているもの（オープン中）のみ、ポジションと言える

（３）MQL4 と MQL5 の「アクティブと履歴」情報の扱い

	ｱｸﾃｨﾌﾞ（ﾄﾚｰﾄﾞ中）ﾄﾚｰﾄﾞ情報	決済済み（履歴）ﾄﾚｰﾄﾞ情報	備考
MQL4	・OrderSelect()の設定により「保有中と待機中」オーダーを選択でき、次ｽﾃｯﾌﾟで、詳細情報の入手が可能 例　①OrderSelect() ②OrderOpenPrice() ・その他で、ﾀﾞｲﾚｸﾄに入手可能なのは OrdersTotal()のみ	・OrderSelect()の設定により「履歴一覧」からオーダーを選択でき、次ｽﾃｯﾌﾟで、詳細情報の入手が可能 （決済済みとキャンセルしたオーダー） 例　①OrderSelect() ②OrderClosePrice() ・その他でﾀﾞｲﾚｸﾄに入手可能なのは OrdersHistoryTotal()のみ	・**「ｱｸﾃｨﾌﾞ」と「履歴（ﾋｽﾄﾘｰ）」の区別なく常時ｱｸｾｽ可能。** OredrSelect はその中から情報を選択する。
MQL5	・詳細手順（例）は、本稿「Ｐ５以降」参照 ・ﾀﾞｲﾚｸﾄに入手可能なﾃﾞｰﾀは OrdersTotal()、PositionsTotal()のみ！	・詳細手順（例）は、本稿「Ｐ７以降」参照 ・ﾀﾞｲﾚｸﾄに入手可能なﾃﾞｰﾀは無し？！	詳細ﾃﾞｰﾀは、必要に際して、一旦、**ｻｰﾊﾞｰからｺﾋﾟｰする手順**が必須

（４）MQL5；データ入手の手順（オーダー等のトータル数）・・「トータル数」等を調べる

（ターミナル（MT5 ；MQL5）側から、サーバー上の情報を入手するステップ）

表３

	ｻｰﾊﾞｰから直接ﾃﾞｰﾀｰ入手		
Orders （注文）	int OrdersTotal()	―	―
Positions	int PositionsTotal()	―	―
		＜ステップ１＞ ｻｰﾊﾞｰから指定「履歴期間」ﾃﾞｰﾀをｷｬｯｼｭにｺﾋﾟｰする	＜ステップ２＞ ｷｬｯｼｭ(cache)からｺﾋﾟｰされたﾃﾞｰﾀ範囲で読取る
Deals (Historical)	―	bool HistorySelect(start,end)	int HistoryDealsTotal()
Historycal orders	―	bool HistorySelect(start,end)	int HistoryOrdersTotal()

※Historical なデータは、期間を指定してトレード・サーバーから、キャッシュにコピーしてから読取ることが可能となる。

※Historical なデータでの「トータル数」とは、飽くまで「キャッシュ」にコピーした履歴期間の中での「トータル数」である。

３．MQL5；データ入手の詳細手順（例）・・「詳細情報」を調べる

※小生は、以下の解説にある「type_property」の詳細については、未だ調査していません。
（　コードの構成例として掲載しています、観ればそのままで判る内容もありますよね！　）

（１）アクティブ・（オーダー）データ入手

Orders		
タイプ１：特定ティケット No　のデータ入手		
	［直接指定］→	［詳細情報を入手］
	bool OrderSelect(ticket)	double OrderGetDouble(type_property)
		long　OrderGetInteger(type_property)
		string OrderGetString(type_property)
タイプ２：全データ調査		
	［ティケットを入手／キャッシュへコピー］→	［詳細情報を入手］
int OrdersTotal()	ulong OrderGetTicket(index) < 0<= index <OrdersTotal() >	double OrderGetDouble(type_property)
		long　OrderGetInteger(type_property)
		string OrderGetString(type_property)

コード構成（例）；
＜タイプ１＞

```
bool selected=OrderSelect(ticket); //ティケットＮＯでオーダーを選択
if(selected)
  {
   double price_open=OrderGetDouble(ORDER_PRICE_OPEN);
   datetime time_setup=OrderGetInteger(ORDER_TIME_SETUP);
   string symbol=OrderGetString(ORDER_SYMBOL);
         PrintFormat("Order #%d for %s was set at %s",ticket,symbol,TimeToString(time_setup));
  }else{
         PrintFormat("Error selecting order with ticket %d. Error %d",ticket, GetLastError());
  }
```

＜タイプ２＞

```
input long my_magic=555;

void OnStart()
  {

   int orders=OrdersTotal();  //  オーダーのトータル数を得る
//  オーダーリストをスキャンする
   for(int i=0;i<orders;i++)
     {
      ResetLastError();
      //  オーダーをリストＮｏ順に、キャッシュにコピーする
      ulong ticket=OrderGetTicket(i);
      if(ticket!=0)// オーダーのコピーが成功したら、下記を実施
        {
         double price_open  =OrderGetDouble(ORDER_PRICE_OPEN);
         datetime time_setup=OrderGetInteger(ORDER_TIME_SETUP);
         string symbol      =OrderGetString(ORDER_SYMBOL);
         long magic_number  =OrderGetInteger(ORDER_MAGIC);
         if(magic_number==my_magic)
           {        //本スクリプト（又はＥＡ）のマジックＮｏであったときの処理を記述        }
          PrintFormat("Order #%d for %s was set out %s, ORDER_MAGIC=%d",
                                    ticket,symbol,TimeToString(time_setup),magic_number);

        }else{
          PrintFormat("Error when obtaining an order from the list to the cache. Error code: %d",
                                                                    GetLastError());
        }

     }
  }
```

（2）アクティブ・（ポジション）データ入手

Positions		
タイプ１：特定為替ペアのみデータ入手		
	［キャッシュへコピー］→	［詳細情報を入手］
	bool PositionSelect(symbol)	double PositionGetDouble(type_property)
		long PositionGetInteger(type_property)
		string PositionGetString(type_property)
タイプ２：全データ調査		
	［キャッシュへコピー］→	［詳細情報を入手］
int PositionsTotal()	string PositionGetSymbol(index) < 0<= index <PositionsTotal() >	double PositionGetDouble(type_property)
		long PositionGetInteger(type_property)
		string PositionGetString(type_property)

※「 PositionGetTicket(index) 」関数は存在しない！！

コード構成（例）；

＜タイプ１＞

```
string symbol=Symbol();                  //  シンボル（為替ペア）を指定
bool selected=PositionSelect(symbol);    //  データをキャッシュへコピー
if(selected) // もしポジションが存在したら、下記を実施
  {
   long pos_id            =PositionGetInteger(POSITION_IDENTIFIER);
   double price           =PositionGetDouble(POSITION_PRICE_OPEN);
   ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE);
   long pos_magic         =PositionGetInteger(POSITION_MAGIC);
   string comment         =PositionGetString(POSITION_COMMENT);
    PrintFormat("Position #%d by %s: POSITION_MAGIC=%d, price=%G, type=%s, commentary=%s",
       pos_id, symbol, pos_magic, price,EnumToString(type), comment);
  }else{
    PrintFormat("Unsuccessful selection of the position by the symbol %s.
                 Error",symbol,GetLastError());
  }
```

※原文を生かしたい部分は翻訳せずに載せた。（要は、微妙な表現なので、的確に翻訳する自信が無いだけ！）

＜タイプ２＞

```
#property script_show_inputs
input long my_magic=555;
void OnStart()
  {
   int positions=PositionsTotal();     //  ポジション総数を得る
   for(int i=0;i<positions;i++)        //  オーダー（ポジション）リストをスキャンする
     {
      ResetLastError();
      //  ポジションをリストＮｏ順に、キャッシュにコピーする
      string symbol=PositionGetSymbol(i);//  ポジションが開かれたシンボルを確認
      if(symbol!="") // ポジションのコピーが成功したら、下記を実施
        {
         long pos_id            =PositionGetInteger(POSITION_IDENTIFIER);
         double price           =PositionGetDouble(POSITION_PRICE_OPEN);
         ENUM_POSITION_TYPE type=(ENUM_POSITION_TYPE)PositionGetInteger(POSITION_TYPE);
         long pos_magic         =PositionGetInteger(POSITION_MAGIC);
         string comment         =PositionGetString(POSITION_COMMENT);
         if(pos_magic==my_magic)
           {      //本スクリプト（又はＥＡ）のマジックＮｏであったときの処理を記述       }
        PrintFormat("Position #%d by %s: POSITION_MAGIC=%d, price=%G, type=%s, commentary=%s",
                    pos_id,symbol,pos_magic,price,EnumToString(type),comment);
        }else{
        PrintFormat("Error when receiving into the cache the position with index %d."+
                    " Error code: %d", i, GetLastError());
        }
     }
  }
```

（３）ヒストリー・（オーダー）データ入手

Historical Orders		
タイプ１：最新データのみ入手		
［キャッシュへコピー］→	［最新のティケットを入手］→	［詳細情報を入手］
bool HistorySelect(start,end)	①int N=HistoryOrdersTotal()	double HistoryOrderGetDouble(ticket,type_property)
	②HistoryOrderGetTicket(N-1)	long HistoryOrderGetInteger(ticket,type_property)
		string HistoryOrderGetString(ticket,type_property)
タイプ２：全データ調査		
［キャッシュへコピー］→	［ティケットを入手］→	［詳細情報を入手］
bool HistorySelect(start,end)	①int HistoryOrdersTotal()	double HistoryOrderGetDouble(ticket,type_property)
	②ulong HistoryOrderGetTicket(index)	long HistoryOrderGetInteger(ticket,type_property)
	< 0<= index <HistoryOrdersTotal() >	string HistoryOrderGetString(ticket,type_property)

コード構成（例）；

＜タイプ１＞・・最新データのみ、入手する場合

```
//  データを入手する履歴期間を決める
   datetime end=TimeCurrent();                        //  現在のサーバー時間
   datetime start=end-PeriodSeconds(PERIOD_D1);       //  １日前の履歴までを指定
   HistorySelect(start,end);                          //  履歴データをキャッシュにコピー
   int history_orders=HistoryOrdersTotal();           //  コピーした履歴データ中のオーダー数を入手
//  履歴リスト中から、最新オーダーのティケットを入手
   ulong order_ticket=HistoryOrderGetTicket(history_orders-1);
   if(order_ticket>0) //  最新オーダーのティケットが存在したら、下記を実施
     {
      //  最新オーダーの情報を入手
      ENUM_ORDER_STATE state=(ENUM_ORDER_STATE)HistoryOrderGetInteger(order_ticket,ORDER_STATE);
      long order_magic      =HistoryOrderGetInteger(order_ticket,ORDER_MAGIC);
      long pos_ID           =HistoryOrderGetInteger(order_ticket,ORDER_POSITION_ID);
      PrintFormat("ｵｰﾀﾞｰ･ﾃｨｹｯﾄ #%d: ORDER_MAGIC=#%d, ORDER_STATE=%d, ｵｰﾀﾞｰの POSITION_ID=%d",
                  order_ticket,order_magic,EnumToString(state),pos_ID);

     }else{

      PrintFormat("In total, in the history of %d orders, we couldn't select the order"+
                  " with the index %d. Error %d",history_orders,history_orders-1,GetLastError());

     }
```

※原文を生かしたい部分は翻訳せずに載せた。（要は、微妙な表現なので、的確に翻訳する自信が無いだけ！）

＜タイプ２＞・・全てのデータを調査する

```
#property script_show_inputs
  ・・・・・
input long my_magic=999;
  ・・・・・
void OnStart()
  {
//  データを入手する履歴期間を決める
   datetime end=TimeCurrent();                        //  現在のサーバー時間
   datetime start=end-PeriodSeconds(PERIOD_D1);       //  １日前の履歴までを指定
   HistorySelect(start,end);                          //  履歴データをキャッシュにコピー
   int history_orders=HistoryOrdersTotal();           //  コピーした履歴データ中のオーダー数を入手
//  履歴リスト中の、全てのオーダー･データを調査
   for(int i=0;i<history_orders;i++)
     {
      //  各オーダーのティケットを入手
      ulong order_ticket=HistoryOrderGetTicket(i);
      if(order_ticket>0) //  ティケットが存在したら、そのデータを入手する          {
         datetime time_done=HistoryOrderGetInteger(order_ticket,ORDER_TIME_DONE);
         long order_magic  =HistoryOrderGetInteger(order_ticket,ORDER_MAGIC);
         long pos_ID       =HistoryOrderGetInteger(order_ticket,ORDER_POSITION_ID);
          //  本スクリプト（又はＥＡ）のマジックＮｏのオーダーがあるかを確認
         if(order_magic==my_magic)
           {         //  本スクリプト（又はＥＡ）のマジックＮｏであったときの処理を記述      }
         PrintFormat("ｵｰﾀﾞｰ･ﾃｨｹｯﾄ #%d: ORDER_MAGIC=#%d, オーダー約定時間 %s, ｵｰﾀﾞｰの POSITION_ID=%d",
                     order_ticket,order_magic,TimeToString(time_done),pos_ID);
        }else{
         PrintFormat("次のオーダーは有りません。 index_NO= %d. エラー＝ %d",i,GetLastError());
        }
     }
  }
```

（４）ヒストリー・(ディール）データ入手

Deals(Histrical)		
タイプ１：最新データのみ入手		
［キャッシュへコピー］→	［最新のティケットを入手］→	［詳細情報を入手］
bool HistorySelect(start,end)	①int **N**=HistoryDealsTotal()	double HistoryDealGetDouble(ticket,type_property)
	②HistoryDealGetTicket(**N-1**)	lonh　HistoryDealGetInteger(ticket,type_property)
		string HistoryDealGetString(ticket,type_property)
タイプ２：全データ調査		
［キャッシュへコピー］→	［ティケットを入手］→	［詳細情報を入手］
bool HistorySelect(start,end)	①int HistoryDealsTotal()	double HistoryDealGetDouble(ticket,type_property)
	②ulong HistoryDealGetTicket(index)	lonh　HistoryDealGetInteger(ticket,type_property)
	〈 0<= index <HistoryDealsTotal() 〉	string HistoryDealGetString(ticket,type_property)

コード構成（例）；
＜タイプ１＞・・最新データのみ、入手する場合

```
//  データを入手する履歴期間を決める
   datetime end=TimeCurrent();                     //  現在のサーバー時間
   datetime start=end-PeriodSeconds(PERIOD_D1);    //  １日前の履歴までを指定
   HistorySelect(start,end);                       //  履歴データをキャッシュにコピー
   int deals=HistoryDealsTotal();                  //  コピーした履歴データ中のディール数を入手
   ulong deal_ticket=HistoryDealGetTicket(deals-1);//履歴リスト中から最新ディールのティケットを入手
   if(deal_ticket>0) //  最新ディールのティケットが存在したら、下記を実施
     {
      //  最新ディールの情報を入手
      ulong order     =HistoryDealGetInteger(deal_ticket,DEAL_ORDER);
      long order_magic=HistoryDealGetInteger(deal_ticket,DEAL_MAGIC);
      long pos_ID     =HistoryDealGetInteger(deal_ticket,DEAL_POSITION_ID);
PrintFormat("Deal #%d for the order #%d with the ORDER_MAGIC=%d that participated in the position=%d",
                  deals-1,order,order_magic,pos_ID);
     }else{

      PrintFormat("In total, in the history %d of deals, we couldn't select a deal"+
                  " with the index %d. Error %d",deals,deals-1,GetLastError());

     }
```

※原文を生かしたい部分は翻訳せずに載せた。（要は、微妙な表現なので、的確に翻訳する自信が無いだけ！）

＜タイプ２＞・・全てのデータを調査する

```
input long my_magic=111;
・・・・・
void OnStart()
  {
//  データを入手する履歴期間を決める
   datetime end=TimeCurrent();                     //  現在のサーバー時間
   datetime start=end-PeriodSeconds(PERIOD_D1);    //  １日前の履歴までを指定
   HistorySelect(start,end);                       //  履歴データをキャッシュにコピー
   int deals=HistoryDealsTotal();                  //  コピーした履歴データ中のディール数を入手
//  計算用データの準備
   int returns=0;   //ディール数
   double profit=0; //利益額
   double loss=0;   //損失額
//  履歴リスト中の、全てのディール・データを調査
   for(int i=0;i<deals;i++)
     {
      ulong deal_ticket=HistoryDealGetTicket(i);   //  各ディールのティケットを入手
      if(deal_ticket>0) // ディールが存在したら、そのデータを入手する
        {
         string symbol      =HistoryDealGetString(deal_ticket,DEAL_SYMBOL);
         datetime time      =HistoryDealGetInteger(deal_ticket,DEAL_TIME);
         ulong order        =HistoryDealGetInteger(deal_ticket,DEAL_ORDER);
         long order_magic   =HistoryDealGetInteger(deal_ticket,DEAL_MAGIC);
         long pos_ID        =HistoryDealGetInteger(deal_ticket,DEAL_POSITION_ID);
         ENUM_DEAL_ENTRY entry_type=(ENUM_DEAL_ENTRY)HistoryDealGetInteger(deal_ticket,DEAL_ENTRY);
         //  本スクリプト（またはＥＡ）のマジックＮｏのディールがあるかを確認
```

```
        if(order_magic==my_magic)
          {        //  本スクリプト（又はＥＡ）のマジックＮｏであったときの処理を記述      }
        //  各トレード結果の「損益」を計算する
        if(entry_type==DEAL_ENTRY_OUT)
         {
           returns++; //  ディール数を「＋１」する
           //  ディール結果「損益（Profit と Loss）」を入手
           double result=HistoryDealGetDouble(deal_ticket,DEAL_PROFIT);
           if(result>0) profit+=result; //  利益額に加算
           if(result<0) loss+=result;   //  損失額を上乗せ
         }
       }else{
         PrintFormat("次のディールは在りません。index_NO＝ %d. エラー＝ %d",i,GetLastError());
       }
    }
        //  計算結果を表示する
        PrintFormat("ディール総数＝ %d  : Profit=%.2f , Loss= %.2f",returns,profit,loss);
}
```

◎アメンボの疑問！；

？ヒストリー・データ中には、独立した「position」データは無いのか？

・「pos_ID」で「order」と「deal」を纏めて、１つの「position」として構成できるけれど、独立したデータとしては存在しても良さそうだが！

・未だ、資料の読込みが足りないせいか、理解できず。

以　上