



Accuracer Developer's Guide

(c) AidAim Software, 2000-2009

Place your own product logo here and modify the layout of your print manual/PDF:

In Help & Manual, click "Tools" > "Print Manual Designer" and open this manual template to edit it.

Title page 1

Use this page to introduce the product

by

This is "Title Page 1" - you may use this page to introduce your product, show title, author, copyright, company logos, etc.

This page intentionally starts on an odd page, so that it is on the right half of an open book from the readers point of view. This is the reason why the previous page was blank (the previous page is the back side of the cover)

Accuracer Developer's Guide

(c) AidAim Software, 2000-2009

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Printed: August 2009 in (whereever you are located)

Publisher

...enter name...

Managing Editor

...enter name...

Technical Editors

...enter name...

...enter name...

Cover Designer

...enter name...

Team Coordinator

...enter name...

Production

...enter name...

Special thanks to:

All the people who contributed to this document, to mum and dad and grandpa, to my sisters and brothers and mothers in law, to our secretary Kathrin, to the graphic artist who created this great product logo on the cover page (sorry, don't remember your name at the moment but you did a great work), to the pizza service down the street (your daily Capricciosas saved our lives), to the copy shop where this document will be duplicated, and and and...

Last not least, we want to thank EC Software who wrote this great help tool called HELP & MANUAL which printed this document.

Table of Contents

Foreword	I
Part I Accuracer -	2
1	2
2	2
3	5
4 ?	6
5	6
.....	6
.....	6
.....	7
.....	9
.....	9
.....	10
.....	12
.....	14
BLOB-	15
BLOB- Varchar-	16
6	17
.....	17
7 SQL	19
.....	19
.....	20
.....	26
.....	28
.....	28
.....	29
.....	29
AVG.....	30
COUNT.....	30
GROUP_CONCAT	30
MIN.....	31
MAX.....	31
SUM.....	32
.....	32
CURRENT_DATE.....	33
CURRENT_TIME.....	33
CURRENT_TIMESTAMP, NOW AND SYSDATE.....	33
DAY.....	33
DAYNAME.....	34
DAYOFWEEK.....	34
EXTRACT.....	34
HOUR.....	35
MINUTE.....	35
MONTH.....	36
MONTHNAME.....	36

MSECOND.....	36
QUARTER.....	36
SECOND.....	37
TODATE.....	37
TOSTRING.....	38
WEEKDAY.....	40
YEAR.....	40
.....	40
ISNULL.....	40
LASTAUTOINC.....	41
.....	41
ABS Function.....	42
CUMPROD Function.....	42
CUMSUM Function.....	42
SIGN Function.....	43
MOD Function.....	43
FLOOR Function.....	43
CEILING Function.....	44
ROUND Function.....	44
TRUNCATE Function.....	44
POWER Function.....	45
HEX Function.....	45
RANDOM Function.....	45
.....	46
LENGTH.....	46
LOWER.....	46
LTRIM.....	47
POS.....	47
RTRIM.....	47
SUBSTRING.....	48
TRIM.....	48
UPPER.....	48
.....	48
CAST.....	49
TOBLOB.....	49
SELECT	50
INSERT	55
UPDATE	55
DELETE	56
CREATE DATABASE Statement	56
DROP DATABASE Statement	57
CREATE TABLE	58
ALTER TABLE	60
DROP TABLE	62
CREATE INDEX	62
DROP INDEX	63
START TRANSACTION	63
COMMIT	64
ROLLBACK	64
8	65
.....	66
.....	68
9	70

	70
10	72
	72
	BDE	73
	EasyTable	73
	74
	74
11	75
	75
12	77
	BDE	77
	78
	79
	81
Index		83

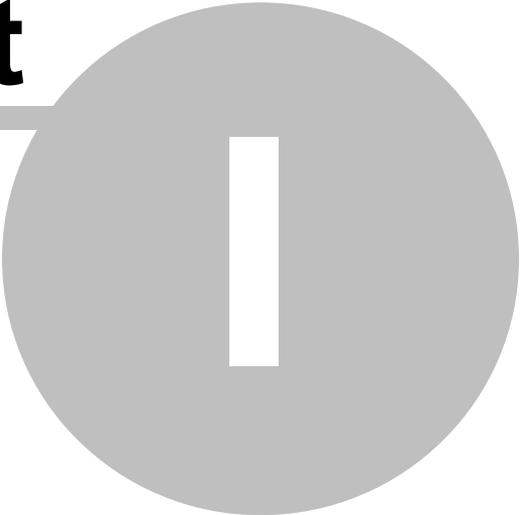
Foreword

This is just another title page
placed between table of contents
and topics

Top Level Intro

This page is printed before a new top-level chapter starts

Part



1 Tuesday, August 04, 2009

1.1

Accuracer:

SQL Delphi, C++ Builder, Kylix ODBC

Accuracer -

SQL.

Main Features:

-
-
-
- BDE DLL- ;
- SQL'92 DDL (SQL) SQL'99)
- (
- ,
- 100%
-
- BLOB Varchar ()
- (SQL DDL)
- BatchMove
- (AES, Blowfish, Twofish, DES . .)
-
- -
- ODBC-
- Windows/Linux: Delphi, C++ Builder Kylix
- , READ COMMITTED
-

<http://www accuracer.com>

<http://www.aidaim.com>

1.2

-
-
- -
- TACRQuery SQL'92, DDL. Accuracer SQL-
- , , ,

```

SELECT.
Accuracer, " SQL" SQL,
( SQL )
'LIKE' . Accuracer
NULL' '%', 'IS NULL' 'IS NOT
. Accuracer , . .
. Accuracer TAccuracer
( NOT NULL).
(
, , ),
SQL & DDL
BatchMove
TACRDatabase RepairDatabase
- READ COMMITTED.
( 740 ),
( , BDE).
Accuracer.
Varchar WideVarchar.
BLOB- . Accuracer
" " , Accuracer,
, PKZip, WinRar, Arj.
CompactDatabase TACRDatabase

```

- B- .
- Accuracer - Delphi C++ Builder.
- . Accuracer Delphi.
- **SQL**
- Accuracer TTable, BLOB- , , **wide-**
- Accuracer QuickReport, DBGrid, DBNavigator, DBImage, DBMemo, DBRichEdit, TDataset - FastReport, DBFlyTreeView
- TTable. TTable, **Key**
- **Range.**
- / . Accuracer
- ImportTable ExportTable. /
- Accuracer
- **Unicode.** , ftWideString.

. Accuracer
 " " ' Accuracer".
 . Builder, Kylix ODBC. - Delphi, C++
 . Blowfish, Twofish, DES . .) (AES,
 . CBC, CFB), - (CTS, OFB,
 . RipeMD128 / RipeMD256) (
 . , , , , ,
 . ,
 . LFSR ,
 . DEC 1, Hagen Reddmann) (

- VCL - Delphi 4,5,6,7, 2005, 2006, 2007 C++ Builder 4,5,6, 2006
- CLX - Kylix 3 Delphi
- ODBC

1.3

/
 : support@aidaim.com.

- : Delphi C++ Builder, , ,
- :
- ()
- ()

1.4

?

www.aidaim.com.

, support@aidaim.com.

sales@aidaim.com.

AidAim

sales@aidaim.com.

1.5

1.5.1

- 1) ACRManager New Database item Database.
- 2) CreateDatabase TACRDatabase. CreateDatabase

1.5.2

InMemory true.

1. Accuracer TACRDatabase DatabaseName
2. DatabaseFileName - Object
Inspector.

1. InMemory true,
DatabaseName,
2. TACRDatabase TACRTable Accuracer
Name
3. TableName

1. DataSource Data Access
page DataSet

- 2. DataSource, TDBGrid, DataSource,
- 3. TACRDatabase True, Connected
- 4. Active True.

1.5.3

```

CreateTable, TAccuracer.
TableName Exists. CreateTable, FieldDefs, IndexDefs,
FieldDefs
FieldDefs TFieldDef,
TFieldDef, Add TFieldDefs, FieldDefs.
Field Name (String) FieldName
Data Type (TFieldType) DataType TFieldType
Size (Word) Size String.
WideString 0.
Required (Boolean) Required (, NOT NULL)
    
```

Advanced FieldDefs

```

Accuracer
FieldDefs,
BLOB- VARCHAR- AdvFieldDefs FieldDefs
FieldDefs FieldDefs.Clear AdvFieldDefs,
AdvFieldDefs ( TACRAdvFieldDef).
    
```

```

IndexDefs
IndexDefs - TIndexDef,
    
```

```

Add TIndexDefs, TIndexDef,
IndexDefs.
:
Index Name (String) Index Name ,
Fields List (String) Fields List ,
Index Options (TIndexOptions) Index Options (Accuracer TIndexOption).

```

TableName

```

[ixPrimary].
CreateTable.
Exists TAccuracer,
CUSTOMER, Delphi,
'DBDemos.adb' CreateTable:

```

```

begin
with MyAccuracer do
begin
TableName:='customer';
with FieldDefs do
begin
Clear;
Add('CustNo',ftAutoInc,0,False);
Add('Company',ftString,30,False);
Add('Addr1',ftString,30,False);
Add('Addr2',ftString,30,False);
Add('City',ftString,15,False);
Add('State',ftString,20,False);
Add('Zip',ftString,10,False);
Add('Country',ftString,20,False);
Add('Phone',ftString,15,False);
Add('FAX',ftString,15,False);
Add('TaxRate',ftFloat,0,False);
Add('Contact',ftString,20,False);
Add('LastInvoiceDate',ftDateTime,0,False);
end;
with IndexDefs do
begin
Clear;
Add('PrimaryKey','CustNo',[ixPrimary]);
Add('ByCompany','Company',[ixCaseInsensitive]);

```

```

end;
if not Exists then
  CreateTable;
end;
end;

```

1.5.4

```

Accuracer
TDBGrid.
Active
MyAccuracer:
begin
  //
  with MyAccuracer do
    begin
      TableName:='customers';
      ReadOnly:=False;
      Open;
    end;
  end;

  ReadOnly
  TableName
  (ReadOnly=False).
  False
  Close.
  TAccuracer
  MyAccuracer:
begin
  MyAccuracer.Close;
end;

```

1.5.5

```

:

```

First
Last
Next
Prior
SetRecNo

(RecNo
)

BOF EOF

Accuracer

CustTable:

```

CustTable.First;           , EOF:=False
while not CustTable.EOF do , EOF      True
begin
    ...
    CustTable.Next; EOF False ; EOF True, Next
end;

```

1.5.6

```

TAccuracer.           Filter,
FilterOptions Filtered. OnFilterRecord Delphi.

```

Filter

Filter,

State

```
table1.Filter := 'State = ' + QuotedStr('CA');
```

Filter ,
Filter

```
table1.Filter := Edit1.Text;
```

```
table1.Filter := 'Married = TRUE';
```

```
table1.Filter := 'State = ' + QuotedStr(Edit1.Text);
```

```

<
>
>=
<=
=
<>
AND
NOT
OR
[NOT] LIKE
IS [NOT] NULL

```

```
(Custno > 1400) AND (Custno < 1500);
```

```

FilterOptions
( )
FilterOptions -

```

```

foCaseInsensitive
foPartialCompare
( . . . (*)
State:

```

```

FilterOptions := [foCaseInsensitive];
Filter := "State" = "CA";

```

Filtered True.

Filtered False.

1.5.7

TAccuracer. - FindFirst, FindLast, FindNext FindPrior.
Locate Lookup,

Find.

Find - :

- 1.
- 2.
- 3.

: FindFirst(), FindLast(), FindNext() FindPrior().

), True. False. (,
Found, True. False.,
Found True. FindNext, False,

Locate

```
Locate
,
Locate
,
CustTable,
Company "Professional Divers, Ltd.":
var
  LocateSuccess: Boolean;
  SearchOptions: TLocateOptions;
begin
  SearchOptions := [loPartialKey];
  LocateSuccess := CustTable.Locate('Company', 'Professional Divers, Ltd.', SearchOptions);
end;

Locate
True
, False
Locate
```

```

        (
            Lookup),
        VarArrayOf.
        :

with CustTable do
    Locate('Company;Contact;Phone', VarArrayOf(['Sight Diver','P']), loPartialKey);

Locate

        Lookup

Lookup
        ,
        ,
        ,
        . Lookup
        ;
        ,
        /
        Lookup
        /
        CustTable,
        Company
        "Professional
        Divers, Ltd.",
        :

var
    LookupResults: Variant;
begin
with CustTable do
    LookupResults := Lookup('Company', 'Professional Divers, Ltd.', 'Company; Contact; Phone');
end;

Lookup
        ,
        , Lookup
        , Lookup
        Null.
        Lookup
        (
            Lookup),
        VarArrayOf.
        :

var
    LookupResults: Variant;
begin
with CustTable do
    LookupResults := Lookup('Company; City', VarArrayOf(['Sight Diver', 'Christiansted']),
        'Company; Addr1; Addr2; State; Zip');

```

end;

Lookup

1.5.8

```

IndexName IndexFieldNames
', ' , '
IndexName
('),
CustomerName:
begin
with MyAccuracer do
begin
IndexName:='CustomerName';
//
end;
end;
',
(
),
First (
),
IndexName,
).
IndexName
IndexFieldNames
IndexFieldNames
IndexFieldNames
CustomerName CustomerName CustomerNo:
begin
with MyAccuracer do
begin
IndexFieldNames:='CustomerName;CustomerNo';
do something
end;
end;
',
',
Accuracer
',

```

1.5.9

BLOB-

```

        BLOB-      Accuracer      TTable.      BLOB-
        BLOB-      ,
        .
        TACRBlobStream      BLOB-      Accuracer.
TACRBlobStream -      ,
  Binary large object (BLOB).
TACRBlobStream
        BLOB-      ,
        .
        BLOB-      ,      TACRBlobStream,
        TACRBlobStream      ,      BLOB-
        TACRBlobStream
        BLOB-      .
        Notes BLOB-

```

```

procedure TForm1.Button2Click(Sender: TObject);
var
  Buffer: PChar;
  MemSize: Integer;
  Stream: TACRBlobStream;
begin
  Stream := TACRBlobStream.Create(MyAccuracer.FieldByName('Notes') as TBlobField,
  bmRead);
  try
    MemSize := Stream.Size;
    Inc(MemSize);
    Buffer := AllocMem(MemSize);
    try
      Stream.Read(Buffer^, MemSize);
      Memo1.SetTextBuf(Buffer);
    finally
      FreeMem(Buffer, MemSize);
    end;
  finally
    Stream.Free;
  end;
end;

```

1.5.10 BLOB- Varchar-

```

Accuracer          -          SQL varchar
ftString. Varchar-          ,
                    ,          ,
                    ,          ,          varchar-
                    ,          BLOB-          .          SQL
                    ,          ,          Varchar -
Char- -          ,          Varchar -
                    .
                    BLOB- Varchar-          ,
                    .

1)          ACRManager          BLOB- Varchar- :
2)          AdvFieldDefs          TACRTable -          ,
TACRAdvFieldDef.
3)          SQL-          -          CREATE TABLE

          : ZLIB, BZIP          PPM. ZLIB -          ZLIB ,
          (~ 1          ) . BZIP          ,          ZLIB
          (          ) ,          (          ) .
PPM -          ,
          . PPM          ,          (100
          ) .
          (          ) 9 (          ) 1
          "          BLOB-          "          BLOB- (          - ) ,
          Varchar.          -100          ,
          ,          (          PPM).
          :
          2          100          (          BZIP          PPM). PPM
          .          BLOB- (          , BZIP -          100          900 , ZLIB -          256
          BLOB-          "          ) , Accuracer
          Varchar-          " (100          ) .
          :          None,

```

```

,
BLOB-
BLOB-
Varchar-, Char-, Memo- BLOB-
,
BLOB-
,
Varchar Char. Memo-
OnFilterRecord,
).
Memo-
Varchar Memo-
Char-
Varchar- 6 30
,
20-24
(24 ) (20 +
)

```

1.6

1.6.1

```

RestructureTable TACRTable.
RestructureTable,
RestructureFieldDefs RestructureIndexDefs properties.
RestructureFieldDefs
TFieldDef, RestructureFieldDefs
RestructureFieldDefs
TFieldDef, Add TFieldDefs, FieldDefs.
Add
Field Name (String) FileName
Data Type DataType
(TFieldType) TFieldType
Size (Word) Size
String String
0.

```

```

        WideString
        Required (Boolean)          Required (Boolean, NOT NULL)

        RestructureIndexDefs
        . The RestructureIndexDefs property is an array of TIndexDef
        objects, each of them containing information about the index to create.
        RestructureIndexDefs
        Add (TIndexDefs, TIndexDef, IndexDefs)
        Index Name (String)          Index Name
        Fields List (String)         Fields List
        Index Options (TIndexOptions) Index Options (TIndexOption)
        Accuracer (Accuracer)
        RestructureTable
        CUSTOMER.DAT,
        RestructureTable:
        Delphi,
        (
        MyAccuracer.Open;
        MyAccuracer.Close;
        with MyAccuracer do
        begin
        modify fields structure
        with RestructureFieldDefs do
        begin
        //
        Add('Customer Name',aftString,300,False);
        //
        Find('Company').Size := 100;
        //
        DeleteFieldDef('Birthday');
        end;
        modify index definitions
        with RestructureIndexDefs do
        begin
        //
        Add('CustomerName_Index','Customer Name',[ixCaseInsensitive]);
        //

```

```

Find('PrimaryKey').Fields := 'Customer ID';
end;
//
RestructureTable;
end;
MyAccuracer.Open;

```

1.7 SQL

1.7.1

Accuracer SQL SQL'92. - SELECT, INSERT, UPDATE, DELETE, CREATE TABLE, ALTER TABLE, DROP TABLE, CREATE INDEX, DROP INDEX. Accuracer TACRQuery, SQL. Accuracer BDE, SQL MEMORY SQL. SQL SQL TACRQuery. SQL TACRQuery. (;).

SQL:

- SELECT
- INSERT
- UPDATE
- DELETE
- CREATE DATABASE
- DROP DATABASE
- CREATE TABLE
- ALTER TABLE
- DROP TABLE
- CREATE INDEX
- DROP INDEX
- START TRANSACTION
- COMMIT
- ROLLBACK

1.7.2

SQL- Accuracer

```
SELECT *
FROM "Detail Parts"
```

```
SELECT DP.PartNo
FROM [Detail Parts] DP
```

SQL Accuracer-

```
SELECT Orders.[Cust No]
FROM Orders
```

```
SELECT C.Name AS CustName
FROM Customer C
WHERE CustName LIKE 'Bill%'
```

SQL-

```
-- -
SELECT * FROM CUSTOMERS
```

/* */

```
SELECT * FROM CUSTOMERS
/* WHERE (Name = 'Mike') */
ORDER BY CustNo
```

SQL Accuracer-

'ABSOLUTE'
, 'ACTION'
, 'ADD'
, 'ALL'
, 'ALLOCATE'
, 'ALTER'
, 'AND'
, 'ANY'
, 'ARE'
, 'AS'
, 'ASC'
, 'ASSERTION'
, 'AT'
, 'AUTHORIZATION'
, 'AVG'
, 'BEGIN'
, 'BETWEEN'
, 'BIT'
, 'BIT_LENGTH'
, 'BOTH'
, 'BY'
, 'CASCADE'
, 'CASCADED'
, 'CASE'
, 'CAST'
, 'CATALOG'
, 'CHAR'
, 'CHARACTER'
, 'CHAR_LENGTH'
, 'CHARACTER_LENGTH'
, 'CHECK'
, 'CLOSE'
, 'COALESCE'
, 'COLLATE'
, 'COLLATION'
, 'COLUMN'
, 'COMMIT'
, 'CONNECT'
, 'CONNECTION'
, 'CONSTRAINT'
, 'CONSTRAINTS'
, 'CONTINUE'
, 'CONVERT'
, 'CORRESPONDING'
, 'COUNT'
, 'CREATE'
, 'CROSS'
, 'CURRENT'
, 'CURRENT_DATE'
, 'CURRENT_TIME'
, 'CURRENT_TIMESTAMP'
, 'CURRENT_USER'
, 'CURSOR'
, 'DATE'
, 'DAY'
, 'DEALLOCATE'

, 'DEC'
, 'DECIMAL'
, 'DECLARE'
, 'DEFAULT'
, 'DEFERRABLE'
, 'DEFERRED'
, 'DELETE'
, 'DESC'
, 'DESCRIBE'
, 'DESCRIPTOR'
, 'DIAGNOSTICS'
, 'DISCONNECT'
, 'DISTINCT'
, 'DOMAIN'
, 'DOUBLE'
, 'DROP'
, 'ELSE'
, 'END'
, 'END-EXEC'
, 'ESCAPE'
, 'EXCEPT'
, 'EXCEPTION'
, 'EXEC'
, 'EXECUTE'
, 'EXISTS'
, 'EXTERNAL'
, 'EXTRACT'
, 'FALSE'
, 'FETCH'
, 'FIRST'
, 'FLOAT'
, 'FOR'
, 'FOREIGN'
, 'FOUND'
, 'FROM'
, 'FULL'
, 'GET'
, 'GLOBAL'
, 'GO'
, 'GOTO'
, 'GRANT'
, 'GROUP'
, 'HEX'
, 'HAVING'
, 'HOUR'
, 'IDENTITY'
, 'IF'
, 'IMMEDIATE'
, 'IN'
, 'INDICATOR'
, 'INITIALLY'
, 'INNER'
, 'INPUT'
, 'INSENSITIVE'
, 'INSERT'
, 'INT'
, 'INTEGER'

, 'INTERSECT'
, 'INTERVAL'
, 'INTO'
, 'IS'
, 'ISNULL'
, 'ISOLATION'
, 'JOIN'
, 'KEY'
, 'LANGUAGE'
, 'LAST'
, 'LEADING'
, 'LEFT'
, 'LEVEL'
, 'LIKE'
, 'LOCAL'
, 'LOWER'
, 'MATCH'
, 'MAX'
, 'MEMORY'
, 'MIME64'
, 'MIN'
, 'MINUS'
, 'MINUTE'
, 'MODULE'
, 'MONTH'
, 'NAMES'
, 'NATIONAL'
, 'NATURAL'
, 'NCHAR'
, 'NEXT'
, 'NO'
, 'NOFLUSH'
, 'NOT'
, 'NULL'
, 'NULLIF'
, 'NUMERIC'
, 'OCTET_LENGTH'
, 'OF'
, 'ON'
, 'ONLY'
, 'OPEN'
, 'OPTION'
, 'OR'
, 'ORDER'
, 'OUTER'
, 'OUTPUT'
, 'OVERLAPS'
, 'PAD'
, 'PARTIAL'
, 'POSITION'
, 'PRECISION'
, 'PREPARE'
, 'PRESERVE'
, 'PRIMARY'
, 'PRIOR'
, 'PRIVILEGES'
, 'PROCEDURE'

, 'PUBLIC'
, 'READ'
, 'REAL'
, 'REFERENCES'
, 'RELATIVE'
, 'RESTRICT'
, 'REVOKE'
, 'RIGHT'
, 'ROLLBACK'
, 'ROWS'
, 'SCHEMA'
, 'SCROLL'
, 'SECOND'
, 'SECTION'
, 'SELECT'
, 'SESSION'
, 'SESSION_USER'
, 'SET'
, 'SIZE'
, 'SMALLINT'
, 'SOME'
, 'SPACE'
, 'SQL'
, 'SQLCODE'
, 'SQLERROR'
, 'SQLSTATE'
, 'START'
, 'SUBSTRING'
, 'SUM'
, 'SYSTEM_USER'
, 'TABLE'
, 'TEMPORARY'
, 'THEN'
, 'TIME'
, 'TIMESTAMP'
, 'TIMEZONE_HOUR'
, 'TIMEZONE_MINUTE'
, 'TO'
, 'TOP'
, 'TRAILING'
, 'TRANSACTION'
, 'TRANSLATE'
, 'TRANSLATION'
, 'TRIM'
, 'TRUE'
, 'UNION'
, 'UNIQUE'
, 'UNKNOWN'
, 'UPDATE'
, 'UPPER'
, 'USAGE'
, 'USER'
, 'USING'
, 'VALUE'
, 'VALUES'
, 'VARCHAR'
, 'VARYING'

```

,VIEW'
,WHEN'
,WHENEVER'
,WHERE'
,WITH'
,WORK'
,WRITE'
,YEAR'
,ZONE'
,PASSWORD'           //          DDL
,BLOBBLOCKSIZE'     //          DDL
,BLOBCOMPRESSIONMODE' //        DDL
,BLOBCOMPRESSIONALGORITHM' //      DDL
,LASTAUTOINC'       //          DDL
,MODIFY'            // alter table blablaba modify ...
,NEW'               //      NEW PASSWORD ALTER TABLE
,INDEX'             //      CREATE INDEX ...
,NOCASE'           //      CREATE INDEX ... NOCASE ..
,LTRIM'
,RTRIM'
,POS'
,LENGTH'
,SYSDATE'          //      DateTime
,NOW'
,TOBLOB'           //      TOBLOB
,TODATE'           //      TODATE
,TOSTRING'         //      TOSTRING
,AUTOINDEXES'     //      DDL AutoIndexes
,NOAUTOINDEXES'  //      DDL AutoIndexes
,INCREMENT'
,LASTVALUE'
,MAXVALUE'
,MINVALUE'
,CYCLED'
,NOMAXVALUE'
,NOMINVALUE'
,NOCYCLED'
,INITIALVALUE'
,RENAME'
,QUARTER'
,WEEKDAY'
,DAYOFWEEK'
,DAYNAME'
,MONTHNAME'
,MSECOND'
,ABS'
,CEILING'
,CEIL'
,FLOOR'
,MOD'
,POWER'
,POW'
,RANDOM'
,RAND'
,ROUND'
,SIGN'
,TRUNCATE'

```

```

,TRUNC'
,SHL'
,SHR'
,DATABASE'
,FILE'
,PAGESIZE'
,MAXSESSIONSCOUNT'
,CUMSUM'
,CUMPROD'
,GROUP_CONCAT'

```

1.7.3

Accuracer SQL

-
-
-
-

```

+ ( )
- ( )
* ( )
/( )

```

:

```

SELECT (Price * Quantity) AS Total
FROM Order

```

Name	Syntax	Description
SHL	SHL	
	<<	
SHR	SHR	
	>>	
XOR	XOR	^
		1,
MOD	%, MOD	MOD
AND	&	
OR		

NOT ~

BLOB, Memo, FmtMemo

Graphic.

=, ==	
>	,
<	,
>=	
<=	
<>, !=	í ãî

TRUE, FALSE UNKNOWN. Boolean, Boolean, :

NULL UNKNOWN. WHERE

```
SELECT *
FROM Orders
WHERE (TaxRate > 0)
```

Boolean TRUE, FALSE.

AND, &&	TRUE,	TRUE.
BETWEEN	TRUE,	
IN	TRUE,	
LIKE	TRUE,	
NOT, !		
OR	TRUE,	TRUE.
IS NULL,	TRUE,	UNKNOWN.
= NULL		
IS NOT	FALSE,	UNKNOWN.
NULL,		
<> NULL		
= ""	TRUE,	IS NULL.
<> ""	FALSE,	IS NOT NULL.

```

:

SELECT *
FROM Customer
WHERE (Company LIKE '%Club%')

SELECT *
FROM Orders
WHERE (ShipToCity IS NOT NULL)

SELECT *
FROM Orders
WHERE (TaxRate BETWEEN 0 and 5) AND (AmountPaid > 1)

SELECT *
FROM Orders
WHERE (ShipVIA IN ('UPS', 'DHL'))

```

(II).

(+)

```

:

SELECT (FirstName + ' ' + LastName) AS Name
FROM Customers

```

1.7.4

C++ Pascal.

0xff, \$ff

```

:
drop table test1;
create table test1 (int1 SmallInt, int2 SmallInt);
insert into test1 values (0xff,$ff);
insert into test1 values (0x00,$0f);
insert into test1 values (0x01,$002);
insert into test1 values (0x03,$0002);
select int1, int2, int1 XOR int2 as int3, int2 ^ int1 as int4,
HEX(int1 XOR int2) as str1, HEX(int2 ^ int1,1) as str2, HEX(int2 ^
int1,2) as str3
from test1 order by int1;

```

1.7.5

SQL:

-
-

-
-
-
-

1.7.5.1

(:).
 SQL.
 SELECT, INSERT, UPDATE DELETE.
 :

```

With ACRQuery1 do
begin
  SQL.Clear;
  SQL.Add('INSERT INTO customer_Sort (Company,Address,CustNo)');
  SQL.Add('VALUES (:Company, :Address, :CustNo)');
  Params[0].AsString := 'AidAim Software';
  Params[1].AsString := 'US';
  Params[2].AsInteger := 4;
  ExecSQL;
end;
    
```

```

With ACRQuery1 do
begin
  SQL.Clear;
  SQL.Add('SELECT * FROM orders WHERE PaymentMethod = :PayMethod ');
  ParamByName('PayMethod').AsString := 'Visa';
  Open;
end;
    
```

1.7.5.2

AVG	AVG (
	[DISTINCT]		
)		
COUNT	COUNT (
	[DISTINCT]		
	*)		
GROUP_CO	GROUP_CO	()	
NCAT	NCAT (
	[DISTINCT]		
	[ASC] [DESC]		
	expression [,		
	Separator])		
MIN	MIN (
)		
MAX	MAX (
)		
SUM	SUM (SUM
	[DISTINCT]		
)		

expression

```

        ,
        DISTINCT,
        NULL ( )
ASC or DESC
        (ASC)
        (DESC). DESC
    
```

Separator

GROUP_CONCAT GROUP BY

Examples:

```

SELECT GROUP_CONCAT(name) FROM test
SELECT GROUP_CONCAT(DESC name) FROM test
SELECT num, GROUP_CONCAT(DISTINCT DESC name, "; ") FROM test GROUP BY
num
    
```

1.7.5.2.4 MIN

GROUP_CONCAT Function

```

MIN (
    :
    )
    :
    , Blob-
    :
    :
SELECT MIN(OrderNo) FROM Orders
SELECT MIN(Company) FROM Customer
    
```

1.7.5.2.5 MAX

```

MAX (
    :
    )
    :
    , Blob-
    :
    :
SELECT MAX(SaleDate) FROM Orders
    
```

1.7.5.2.6 SUM

SUM

```
SUM ( [DISTINCT] )
```

DISTINCT SUM

```
SELECT SUM(AmountPaid) FROM Orders WHERE PaymentMethod='Visa'
SELECT SUM(DISTINCT EmpNo) FROM Orders
```

1.7.5.3

```
CURRENT_DATE CURRENT_DATE
CURRENT_TIME CURRENT_TIME
CURRENT_TIMESTAMP CURRENT_TIMESTAMP
TAMP P
DAY DAY ( ) ( : 1 - 31),
DAYNAME DAYNAME ( ( ),
DAYOFWEEK DAYOFWEEK ( , 2 - , ..., 7 - : 1 - ),
EXTRACT EXTRACT ( FROM ) , , , , ,
HOUR HOUR ( ) ( : 0 - 23),
MINUTE MINUTE ( ) ( : 0 - 59),
MONTH MONTH ( ) ( : 1 - 12),
MONTHNAME MONTHNAME ( ( ),
MSECOND MSECOND ( ) 999), ( : 0 -
NOW NOW
QUARTER QUARTER ( ( : 1 - 4),
SECOND SECOND ( ( : 0 - 59),
SYSDATE SYSDATE
TODATE TODATE( StringValue,
DateFormat )
TOSTRING TOSTRING( DateValue,
DateFormat )
```

WEEKDAY WEEKDAY (, 2 - (, ..., 7 - : 1 -))
 YEAR YEAR () () ,

1.7.5.3.1 CURRENT_DATE

```

:
CURRENT_DATE

:
SELECT LastInvoiceDate, CURRENT_DATE as CurDate
FROM Customer
WHERE LastInvoiceDate < NOW
    
```

1.7.5.3.2 CURRENT_TIME

```

:
CURRENT_TIME

:
SELECT LastInvoiceDate, CURRENT_TIME as CurTime
FROM Customer
WHERE LastInvoiceDate < NOW
    
```

1.7.5.3.3 CURRENT_TIMESTAMP, NOW AND SYSDATE

```

:
CURRENT_TIMESTAMP
NOW
SYSDATE

:
SELECT LastInvoiceDate, NOW as CurDate
FROM Customer
WHERE LastInvoiceDate < NOW
    
```

1.7.5.3.4 DAY

```

( : 1 - 31),

:
    
```

```

DAY (
    :
    , - ,
    :
SELECT DAY(LastInvoiceDate) FROM Customer

```

1.7.5.3.5 DAYNAME

```

(
    :
DAYNAME (
    :
    , - ,
    :
SELECT DAYNAME(LastInvoiceDate) FROM Customer

```

1.7.5.3.6 DAYOFWEEK

```

( :1 - , 2 - , ..., 7 - ),
    :
DAYOFWEEK (
    :
    , - ,
    :
SELECT DAYOFWEEK(LastInvoiceDate) FROM Customer

```

1.7.5.3.7 EXTRACT

```

, , , , , , - ,
    :
EXTRACT ( FROM )
EXTRACT ( , )
    :
    :
```

```

DAY ( : 1 - 31),
DAYNAME ( ),
DAYOFWEEK ( : 1 - , 2 - , ..., 7 - ),
HOUR ( : 0 - 23),
MINUTE ( : 0 - 59),
MONTH ( : 1 - 12),
MONTHNAME ( ),
MSECOND ( : 0 - 999),
QUARTER ( : 1 - 4),
SECOND ( : 0 - 59),
WEEKDAY ( : 1 - , 2 - , ..., 7 - ),
YEAR ( ),

```

```

:
SELECT EXTRACT(YEAR FROM LastInvoiceDate) FROM Customer
SELECT EXTRACT(MONTH, LastInvoiceDate) FROM Customer

```

1.7.5.3.8 HOUR

```

( : 0 - 23),
:
HOUR ( )
:
:
SELECT HOUR(LastInvoiceDate) FROM Customer

```

1.7.5.3.9 MINUTE

```

( : 0 - 59),
:
MINUTE ( expression)
:
:

```

```

:
SELECT MINUTE(LastInvoiceDate) FROM Customer

```

1.7.5.3.10 MONTH

```

( : 1 - 12),

:
MONTH ( )

```

```

:
SELECT MONTH(LastInvoiceDate) FROM Customer

```

1.7.5.3.11 MONTHNAME

```

( ),

:
MONTHNAME ( )

```

```

:
SELECT MONTHNAME(LastInvoiceDate) FROM Customer

```

1.7.5.3.12 MSECOND

```

( : 0 - 999),

:
MSECOND ( )

```

```

:
SELECT MSECOND(LastInvoiceDate) FROM Customer

```

1.7.5.3.13 QUARTER

```

( : 1 - 4),

:
QUARTER ( )

```

```

:
, - ,
:
SELECT QUARTER(LastInvoiceDate) FROM Customer

```

1.7.5.3.14 SECOND

```

( : 0 - 59),
:
SECOND ( )
:
, - ,
:
SELECT SECOND(LastInvoiceDate) FROM Customer

```

1.7.5.3.15 TODATE

```

:
TODATE( StringValue, DateFormat )
:
StringValue
DateFormat ,
, StringValue.
DateFormat ,
( , "d")
( , "/" )

```

```

-
/
.
;
:
;
'TEXT'
YYYY (0000-9999)
or

```

YEAR
 YY (00-99)
 Q (1-4). 1 - , 2 - , 3 - , 4 - ,

MONTH (-).
 MON (-).
 MM (01-12).
 M (1-12).
 RM (I - XII).
 DDD (1-366) .
 DD (1-31) .
 D (1-31) .
 DAY (-).
 DY (-).
 DW (1-7)
 HH (01-12).
 HH12 (01-24).
 HH24 (01-24).
 H (1-12).
 H12 (1-24).
 H24 (1-24).
 NN (00:59).
 N (0:59).
 SS (00:59).
 S (00:59).
 ZZZ (000:999).
 Z (0:999).
 AMPM AM.

```

:
SELECT LastInvoiceDate, NOW as CurDate
FROM Customer
WHERE LastInvoiceDate < TODATE('12/16/2002 11:10:30 am','MM/DD/YYYY
hh:nn:ss ampm')

```

1.7.5.3.16 TOSTRING

```

:
TOSTRING( DateValue, DateFormat )
:
DateValue
DateFormat , - ,
:
:
DateFormat , ( , "d") ,
( , "/" ) ( , )

```

```

-
/
.
,
:
;
'TEXT'
,
YYYY (0000-9999)
or
YEAR
YY (00-99)
Q (1-4). 1 - , 2 - , 3 - , 4 - ,
MONTH ( - ).
MON ( - ).
MM (01-12).
M (1-12).
RM (I - XII).
DDD (1-366)
DD (1-31)
D (1-31)
DAY ( - ).
DY ( - ).
DW (1-7)
HH (01-12).
HH12 (01-24).
HH24 (01-24).
H (1-12).
H12 (1-24).
H24 (1-24).
NN (00:59).
N (0:59).
SS (00:59).
S (00:59).
ZZZ (000:999).
Z (0:999).
AMPM AM.

```

```

:
SELECT TOSTRING>LastInvoiceDate, "'Today is' mm/dd/yyyy hh24:nn:ss:zzz '
Wow !!!'" Formated_Date, LastInvoiceDate
FROM Customer

```

1.7.5.3.17 WEEKDAY

```

        ( :1 - , 2 - , ..., 7 - ),
        :
WEEKDAY ( )
        :
        , - ,
        :
SELECT WEEKDAY(LastInvoiceDate) FROM Customer

```

1.7.5.3.18 YEAR

```

        ( ),
Syntax:
YEAR ( expression )
        :
        , - ,
        :
SELECT YEAR(LastInvoiceDate) FROM Customer

```

1.7.5.4

```

ISNULL ISNULL ( , ,
        [ , true, false,
        ] )
LASTAUTO LASTAUTOINC(
INC - )

```

1.7.5.4.1 ISNULL

```

, true, false,
:
ISNULL ( [ - ] )
:

```

```

CAST

:
SELECT ISNULL(Addr2,'No Address') FROM Customer
SELECT * FROM Customer WHERE ISNULL(Addr2)
SELECT SUM(ISNULL(AmountPaid,CAST(10.0,CURRENCY))) FROM Orders
    
```

1.7.5.4.2 LASTAUTOINC

```

LASTAUTOINC

:
LASTAUTOINC( _ , _ )

:
- ,
- ,
..
:
INSERT INTO Employee (Name,DeptID)
VALUES ('John Smith',LASTAUTOINC( Department, ID ))
    
```

1.7.5.5

Name	Syntax	Description
ABS	ABS (x)	x
SIGN	SIGN (x)	-1, 0, 1 (x)
MOD	MOD (x, y)	x % y (x, y)
FLOOR	FLOOR (x)	x.
CEILING	CEILING (x) or CEIL (x)	x.
CUMPRO	CUMPROD(x D)	.
CUMSUM	CUMSUM (x)	.
ROUND	ROUND (x) or ROUND (x, d)	d.
TRUNCA TE	TRUNCATE (x, d) or TRUNC (x, d)	X, D = 0,
POWER	POWER (x, y)	x y.

) or POW (x,
 y)
 HEX HEX(X [, X - , X
 MODE]) , X
 ,
 . MODE: 0 -
 (, 1 - C++
 Pascal (\$FF), 2 -
 (0xff).
 RAND, RAND () or
 RANDOM RAND (n) , n , 0 <= X < N. 0

1.7.5.5.1 ABS Function

Returns the absolute value of x.

Syntax;
 ABS (x)

Arguments:

x
 Is an expression of the numeric data type.

Examples:

SELECT abs(int1) as int_val, abs(float1) as float_Val from test1

1.7.5.5.2 CUMPROD Function

Syntax;
 CUMPROD(x)

Arguments:

x

Examples:

SELECT Cost, CUMPROD(Cost) FROM Orders

1.7.5.5.3 CUMSUM Function

Syntax;
 CUMSUM (x)

Arguments:

x

Examples:

```
SELECT Cost,CUMSUM(Cost) FROM Orders
```

1.7.5.5.4 SIGN Function

Returns the sign of input x as -1, 0, or 1 (negative, zero, or positive respectively).

Syntax;

```
SIGN ( x )
```

Arguments:

x

Is an expression of the numeric data type.

Examples:

```
SELECT sign(int1) as int_val, sign(float1) as float_Val from test1
```

1.7.5.5.5 MOD Function

Returns the integer remainder of x divided by y (the same as x%y).

Syntax;

```
MOD ( x )
```

Arguments:

x

Is an expression of the numeric data type.

Examples:

```
SELECT int1 % int2 as int_Val2, int1 MOD int2 as int_Val3 from test1
```

1.7.5.5.6 FLOOR Function

Returns the largest integer value that is less than or equal to x.

Syntax;

```
FLOOR ( x )
```

Arguments:

x

Is an expression of the numeric data type.

Examples:

```
SELECT FLOOR(int1) as int_val, FLOOR(float1) as float_Val from test1
```

1.7.5.5.7 CEILING Function

Returns the absolute value of x .

Syntax;
CEILING (x)

Arguments:
 x
Is an expression of the numeric data type.

Examples:
SELECT CEIL(int1) as int_val, CEILING(float1) as float_Val from test1

1.7.5.5.8 ROUND Function

Returns the value of x rounded to the nearest whole integer or to the number of decimal places specified by the value d .

Syntax;
ROUND (x) or
ROUND (x , d)

Arguments:
 x
Is an expression of the numeric data type.
 d
Is an expression of the integer data type.

Examples:
SELECT float1, ROUND(float1) as float_Val1, ROUND(float1,1) as float_Val2, ROUND(float1,2) as float_Val3 from test1

1.7.5.5.9 TRUNCATE Function

Returns the number X , truncated to D decimals. If D is 0, the result will have no decimal point or fractional part.

Syntax;
TRUNCATE (x)

Arguments:
 x
Is an expression of the numeric data type.
 d
Is an expression of the integer data type.

Examples:
SELECT float1, TRUNCATE(float1) as float_val1, TRUNC(float1,1) as float_val2, TRUNC(float1,2) as float_val3 from test1

1.7.5.5.10 POWER Function

Returns the value of x raised to the power of y .

Syntax;

POWER (x , y) or
POW (x , y)

Arguments:

x , y
Are an expressions of the numeric data type.

Examples:

```
SELECT POWER(int1,int2) as exp1, POW(float1,int2) as exp2 from test1
```

1.7.5.5.11 HEX Function

If X is a number, returns a string representation of the hexadecimal value of X , where X is integer.
If X is a string, returns a hexadecimal string of X where each character in X is converted to 2 hexadecimal digits.

Syntax;

HEX (X [, *MODE*])

Arguments:

x
Is an expression of the numeric data type.

MODE

0 - just convert to hex (default),
1 - Pascal hex style (\$FF),
2 - C++ hex style (0xff).

Examples:

```
select HEX(int1 XOR int2) as str1, HEX(int2 ^ int1,1) as str2, HEX(int2 ^ int1,2) from test1
```

1.7.5.5.12 RANDOM Function

Returns a random floating-point value in the range 0 to 1.0. If an integer argument n is specified, it is used as maximum value and result will be integer X , where $0 \leq X < n$.

Syntax;

RAND () or
RAND (n) or
RANDOM () or
RANDOM (n)

Arguments:

n
Is an expression of the integer data type.

Examples:

```
select test1.*, RAND(100000) as rnd from memory test1 order by num
desc,id
select test1.*, RANDOM as rnd from memory test1 order by num desc,id
```

1.7.5.6

```
LENGTH LENGTH (          )
LOWER LOWER (          )
LTRIM  LTRIM (          )
POS    POS (          ,          Pos
RTRIM  RTRIM (          )
SUBSTRSUBSTRING (
NG      ,          [,
      ] )
TRIM   TRIM (          )
UPPER  UPPER (          )
```

1.7.5.6.1 LENGTH

```
LENGTH (          )
:
:
:
SELECT * FROM Customer
WHERE LENGTH(Company) > 5
```

1.7.5.6.2 LOWER

```
LOWER (          )
:
:
```

```

:
SELECT LOWER(Company) FROM Customer

```

1.7.5.6.3 LTRIM

Syntax:

```
LTRIM (      )
```

```

:
```

```

:
SELECT LTRIM(Company) FROM Customer

```

1.7.5.6.4 POS

Pos

```

:
POS (      ,      )
```

```

:
```

```

:
SELECT * FROM Customer
WHERE Pos('Blue',Company) > 0

```

1.7.5.6.5 RTRIM

```
RTRIM (      )
```

```

:
```

```

:
SELECT RTRIM(Company) FROM Customer

```

1.7.5.6.6 SUBSTRING

```
SUBSTRING ( , [ , ] )
```

```
SELECT SUBSTRING(Company, 2, 5)  
FROM Customer
```

1.7.5.6.7 TRIM

Syntax:
TRIM ()

```
SELECT TRIM(Company) FROM Customer
```

1.7.5.6.8 UPPER

Syntax:
UPPER ()

```
SELECT UPPER(Company) FROM Customer
```

1.7.5.7

CAST CAST(,)
 TOBLOB TOBLOB([]) BLOB.

1.7.5.7.1 CAST

CAST(: , -)

:

-

CAST

AutoInc		32-	
BCD			
Currency			
Date			
DateTime			
Float			
Integer	32-		
LargeInt	64-		
Logical			
SmallInt	16-		
String			(2^32)
Time			
WideString			Unicode (
Word	2^32)	
	16-		

1.7.5.7.2 TOBLOB

BLOB.

Syntax:

TOBLOB([])

:

, BLOB-

MIME64 - MIME64 ()

HEX -
 MIME64 (

```

).
:
INSERT INTO jpeg VALUES (
  'AidAim',
  TOBLOB ('QWlkQWltIFNvZnR3YXJlDQpIZXJlIHRvIEh1bHANCg==',MIME64),
  NULL, 1);

```

1.7.6 SELECT

SELECT

```

SELECT [DISTINCT | ALL] [TOP n[,
* | [AS | ], [ ...]
[INTO ]
FROM [AS ] [PASSWORD
' ]
[[[NATURAL] [INNER | [LEFT | RIGHT | FULL] OUTER JOIN] - - [AS
| ]
[ON ] | USING ( - )]
[WHERE ]
[GROUP BY - ]
[HAVING ]
[ORDER BY - ]
[UNION [ALL] [CORRESPONDING [BY ( - )]] SELECT...]]
[EXCEPT | MINUS [ALL] [CORRESPONDING [BY ( - )]] SELECT...]]
[INTERSECT [ALL] [CORRESPONDING [BY ( - )]] SELECT...]]

```

SELECT

```

ALL ( )
DISTINCT

```

Customer.

```
SELECT DISTINCT Contact FROM Customer
```

```
TOP , n
```

```
n
INTO ,
```

```
SELECT.
```

```
FROM
WHERE
GROUP BY
SELECT.
```

```
SELECT CustNo, SUM(ItemsTotal)
FROM Orders
GROUP BY CustNo
```

```
HAVING
```

ORDER BY

```

:
1) ORDER BY < _ >
or
2) ORDER BY INDEX _
< _ > ::= [ _ .] _ [ASC | DESC] [NOCASE]
ORDER BY
1
ASC
DESC
NOCASE
Customers
:

```

```

SELECT CustNo, Contact, Company, City, Country
INTO NewCustomer
FROM Customer
ORDER BY Country DESC, City DESC NOCASE, Company NOCASE, Contact

```

2

```

:
SELECT * from Customer_findKey ORDER BY INDEX ByCompany

```

Join, Natural and Using

```

FROM
:
5
- 10
50
FROM _ _ , _ _ [, _ _ ...]
:
SELECT * FROM Members,Departments

```

(LEFT, RIGHT FULL).

Accuracer -

, Accuracer

1) ON:
 FROM [INNER | LEFT | RIGHT | FULL] JOIN
 ON
 [[INNER | LEFT | RIGHT | FULL] JOIN ON ...]

2) Using (
):
 FROM [INNER | LEFT | RIGHT | FULL] JOIN USING
 ([, ...])
 [[INNER | LEFT | RIGHT | FULL] JOIN USING
 ([, ...])...]

3)
 FROM NATURAL [INNER | LEFT | RIGHT | FULL] JOIN
 [NATURAL [INNER | LEFT | RIGHT | FULL] JOIN ...]

```
SELECT Contact, Customer.CustNo, Company, Orders.OrderNo, Orders.CustNo
FROM Customer INNER JOIN Orders
ON (Customer.CustNo = Orders.CustNo)
WHERE Contact LIKE 'E%'
ORDER BY Contact,Orders.CustNo,Orders.OrderNo
```

```
SELECT Contact, Customer.CustNo, Company, Orders.OrderNo, Orders.CustNo
FROM Customer INNER JOIN Orders Using (CustNo)
ORDER BY Contact,Orders.CustNo,Orders.OrderNo
```

```
SELECT cb.*
FROM Customer_Base cb NATURAL INNER JOIN Customer_Base
```

```
SELECT Contact, Customer.CustNo, Company, Orders.OrderNo, Orders.CustNo
FROM Customer NATURAL LEFT JOIN Orders
WHERE State IS NOT NULL
ORDER BY Contact,Orders.CustNo,Orders.OrderNo
```

```
SELECT Contact, Customer.CustNo, Company, Orders.OrderNo, Orders.CustNo
FROM Customer NATURAL RIGHT JOIN Orders
WHERE State IS NOT NULL
ORDER BY Contact,Orders.CustNo,Orders.OrderNo
```

```
SELECT Contact, Customer.CustNo, Company, Orders.OrderNo, Orders.CustNo
FROM Customer NATURAL FULL JOIN Orders
WHERE State IS NOT NULL
```

ORDER BY Contact,Orders.CustNo,Orders.OrderNo

UNION

[UNION [ALL] [CORRESPONDING [BY (_)]] SELECT...]

UNION:

- CORRESPONDING,
- CORRESPONDING,
-

ALL

```
SELECT Company FROM customer_Base
UNION
SELECT Company FROM customer_Filter
UNION
SELECT Company FROM customer_Range
```

EXCEPT (MINUS)

[EXCEPT [ALL] [CORRESPONDING [BY (_)]] SELECT...]

EXCEPT:

- CORRESPONDING,
- CORRESPONDING,
-

ALL

```
SELECT * FROM customer_Range
EXCEPT CORRESPONDING BY (Company)
SELECT * FROM customer_Filter
```

INTERSECT

```
[INTERSECT [ALL] [CORRESPONDING [BY ( _ )]] SELECT...]
```

INTERSECT:

- CORRESPONDING,
- CORRESPONDING,
-

ALL

```
SELECT * FROM customer_Range
INTERSECT CORRESPONDING BY (Company)
SELECT * FROM customer_Filter
```

WHERE

WHERE

Accuracer

- **Accuracer:**

```
SELECT field1 FROM table1 T1
WHERE T1.field2 = (SELECT MAX(field1) FROM table2 T2 WHERE T2.field2 = T1.field3);
```

Utils\Bin\SQLConsole\SQL\SubQuery.

```
SELECT * from Jpeg
WHERE ID = (SELECT MIN(ID) from jpeg)
```

```
SELECT * FROM orders
WHERE CustNo IN
(SELECT DISTINCT CustNo FROM customer WHERE (Company LIKE 'S%') and (CustNo <
2500))
ORDER BY CustNo
```

```
SELECT Count(*) as ROW_COUNT FROM jpeg
WHERE EXISTS
(SELECT * FROM jpeg WHERE (Name LIKE 'A%'))
```

1.7.7 INSERT

INSERT

```
INSERT INTO [table_name] ([column_name])
VALUES (value) [password 'pass'] [(value)]
```

```
INSERT INTO MEMORY
VALUES (value), (value)
```

INSERT:

```
INSERT INTO Customer (CustNo,Company, City, State, Contact, LastInvoiceDate)
VALUES (5555,'AidAim Software','Phoenix','AZ','Ella Perelman','10/15/2002')
```

VALUES

```
INSERT INTO Customer_Sort
SELECT * FROM Customer_Share
```

1.7.8 UPDATE

UPDATE

```
UPDATE [table_name] SET [column_name] = [Password "password_value"]
[WHERE condition]
```

UPDATE

UPDATE

MEMORY

SET -

UPDATE.

```
SET [column_name] = [value], [column_name] = [value] ...]
```

```

        SET
        WHERE
    WHERE
        WHERE
    :
    UPDATE Members SET FirstName = 'New Name' WHERE ID >= 3

```

1.7.9 DELETE

```

    DELETE
    DELETE FROM [database_name] [table_name] [PASSWORD 'password_string']
    [WHERE clause]
    FROM MEMORY
    FROM
    WHERE
    WHERE
    WHERE
    WHERE
    DELETE FROM Members WHERE ID >= 3

```

1.7.10 CREATE DATABASE Statement

```

    CREATE DATABASE
    CREATE DATABASE
        FILE "file_name" [ PAGESIZE {128..65535} ] [ MAXSESSIONSCOUNT {1..2147483648} ]
        |
        MEMORY database_name

```

```

CREATE DATABASE
FILE MEMORY
        file_name.
        ('          (").
        -          PAGESIZE or MAXSESSIONCOUNT ,
        ,          ,          )
    )

```

1.
 CREATE DATABASE FILE "c:\temp\test.adb" PAGESIZE 8192 MAXSESSIONSCOUNT 10
 -

2.
 CREATE DATABASE MEMORY MemDB1
 - MemDB1.

1.7.11 DROP DATABASE Statement

```

DROP DATABASE
FILE MEMORY
        file_name.
        ('          (").
        ,          ,          )
    )

```

1.
 DROP DATABASE FILE "c:\temp\test.adb"
 - c:\temp\test.adb.

2.
 DROP DATABASE MEMORY MemDB1
 - MemDB1.

1.7.12 CREATE TABLE

CREATE TABLE

```

CREATE TABLE            (
                          [(            )]|
    AutoInc([(           
        [, INCREMENT integer ]
        [, INITIALVALUE integer ]
        [, MAXVALUE integer | NOMAXVALUE ]
        [, MINVALUE integer | NOMINVALUE ]
        [, CYCLED | NOCYCLED ]
    )])

    [ BLOBBLOCKSIZE {1..4294967295} ]
    [ BLOBCOMPRESSIONALGORITHM {NONE | ZLIB | BZIP | PPM} ]
    [ BLOBCOMPRESSIONMODE {0 .. 9} ]

    [ DEFAULT {const | NULL} ]
    [ NOT NULL | NULL ]
    [ PRIMARY [KEY] | UNIQUE [ ASC | DESC ] [ CASE | NOCASE ] ]
    [MINVALUE            | NOMINVALUE ]
    [MAXVALUE            | NOMAXVALUE ]
    [,            ...]

    [, PRIMARY KEY [            ] (            [ ASC | DESC ] [ CASE | NOCASE ]
    [ {,            [ ASC | DESC ] [ CASE | NOCASE ] } ... ) ] ]
    [, FOREIGN KEY [            ] (            [ {,            } ... ] )
    REFERENCES            [MATCH FULL | MATCH PARTIAL]
    [ON DELETE <CASCADE | SET NULL | SET DEFAULT | NO ACTION>]
    [ON UPDATE <CASCADE | SET NULL | SET DEFAULT | NO ACTION>] ] ]
)

```

 = [MEMORY]

CREATE TABLE

MEMORY

 :

	<u>TFieldType</u>
Char, FixedChar	ftFixedChar
Varchar, Varchar2, String	ftString
WideChar, FixedWideChar	ftWideString
WideVarchar, WideString	ftWideString

Shortint, SignedInt8	8-		ftSmallint
Smallint, SignedInt16	16-		ftSmallint
Integer, SignedInt32	32-		ftInteger
Largeint, Int64, SignedInt64	64-		ftLargeint
Byte, UnsignedInt8			ftWord
Word, UnsignedInt16	16-		ftWord
Cardinal, UnsignedInt32	32-		ftLargeint
AutoInc, AutoincInteger	32-		ftAutoinc
AutoIncShortint		8-	ftAutoinc
AutoIncSmallint	16-		ftAutoinc
AutoIncLargeint	64-		ftAutoinc
AutoIncByte			ftAutoinc
AutoIncWord	16-		ftAutoinc
AutoIncCardinal	32-		ftAutoinc
Single			ftFloat
Float, Double			ftFloat
Extended			ftFloat
Boolean, Logical, Bool, Bit			ftBoolean
Currency, Money			ftCurrency
Date			ftDate
Time			ftTime
DateTime		-	ftDateTime
TimeStamp		-	ftTimeStamp
		dbExpress	
Bytes	()	ftBytes
VarBytes	()	ftVarBytes
Blob		Binary Large Object	ftBlob

```

Graphic                               ftGraphic
Memo, Clob                             ftMemo

FormattedMemo,                         ftFmtMemo
FmtMemo

WideMemo, WideClob                     ftMemo
                                         Unicode

, - .
, NOT NULL

BLOB- (BLOB, FmtMemo, Memo, Graphic)
BlobCompressionAlgorithm BlobCompressionMode.
BlobBlockSize BLOB- BLOB-
- 1 , - 100 / BLOB-
:

```

```

CREATE TABLE Test
(
  ID AutoInc PRIMARY KEY,
  Text String(500),
  Numeric Float,
  Money Currency,
  CurrentDate Date,
  Picture Graphic BlobCompressionAlgorithm ZLIB BlobCompressionMode 1
);

```

```

:
CREATE TABLE
, (, , TABLE)
( , , ' )

```

1.7.13 ALTER TABLE

ALTER TABLE

```

ALTER TABLE _ ADD [COLUMN]
(
  _ [( )] [NOT NULL]
  [, _ [( )] [NOT NULL]...]
  [,PRIMARY KEY ( _ [, _ ...])]
)
ALTER TABLE _ ADD
(

```

```

[ PRIMARY KEY [ _ ] ( _ [ ASC | DESC ] [ CASE | NOCASE ]
[ FOREIGN KEY [ _ ] ( _ [ { , _ } ... ]
REFERENCES _ [MATCH FULL | MATCH PARTIAL]
[ON DELETE <CASCADE | SET NULL | SET DEFAULT | NO ACTION>]
[ON UPDATE <CASCADE | SET NULL | SET DEFAULT | NO ACTION>]]
)

ALTER TABLE _ <MODIFY> | <ALTER [COLUMN]> (
_ [(dimensions) |
AutoInc([data_type ]
[ , INCREMENT integer ]
[ , INITIALVALUE integer ]
[ , MAXVALUE integer | NOMAXVALUE ]
[ , MINVALUE integer | NOMINVALUE ]
[ , CYCLED | NOCYCLED ]
)]

[ BLOBBLOCKSIZE {1..4294967295} ]
[ BLOBCOMPRESSIONALGORITHM {NONE | ZLIB | BZIP | PPM} ]
[ BLOBCOMPRESSIONMODE {0 .. 9} ]

[ DEFAULT {const | NULL} | DROP DEFAULT ]
[ NOT NULL | NULL ]
[ PRIMARY [KEY] | UNIQUE [ ASC | DESC ] [ CASE | NOCASE ]]
[MINVALUE | NOMINVALUE ]
[MAXVALUE | NOMAXVALUE ]
)

ALTER TABLE _ DROP [COLUMN]
(
_ [, _ ...]
)

ALTER TABLE _ DROP CONSTRAINT _ [CASCADE | RESTRICT]

ALTER TABLE _ RENAME [COLUMN] _ [ TO ] _

ALTER TABLE _ RENAME TO _
or
RENAME TABLE _ TO _

ALTER TABLE
ADD
ALTER MODIFY
: Accuracer
,
NULL.
DROP
:
```

```
ALTER TABLE Test DROP (Numeric);
```

```
ALTER TABLE Test ADD (NewField WideString(500));
```

```
ALTER TABLE Test DROP CONSTRAINT PK CASCADE
```

```
ALTER TABLE Emp ADD FOREIGN KEY FKDeptID (DeptID) REFERENCES Dept MATCH FULL
ON DELETE CASCADE ON UPDATE SET DEFAULT
```

1.7.14 DROP TABLE

```
DROP TABLE
```

```
DROP TABLE _ [CASCADE | RESTRICT]
```

```
    CASCADE
```

```
    ( , ).
```

```
:
```

```
DROP TABLE Test
```

```
:
```

```
, DROP TABLE
```

1.7.15 CREATE INDEX

```
CREATE INDEX
```

```
CREATE [UNIQUE] INDEX [IF NOT EXISTS] _ ON _
```

```
( _ [ASC | DESC] [CASE | NOCASE]
```

```
 [, _ ...]
```

```
)
```

```
CREATE INDEX
```

```
UNIQUE
```

```
IF NOT EXISTS
```


START TRANSACTION

:

```
START TRANSACTION;
INSERT INTO Table1 (NAME) VALUES('aaa');
UPDATE Table2 SET Field1 = Field1 + 1;
INSERT INTO Table1 (NAME) VALUES('bb');
COMMIT;
```

1.7.18 COMMIT

COMMIT

NOFLUSH

, COMMIT

NOFLUSH , COMMIT

COMMIT

COMMIT [NOFLUSH]

:

```
START TRANSACTION;
INSERT INTO Table1 (NAME) VALUES('aaa');
UPDATE Table2 SET Field1 = Field1 + 1;
INSERT INTO Table1 (NAME) VALUES('bb');
COMMIT NOFLUSH;
```

1.7.19 ROLLBACK

ROLLBACK

,

ROLLBACK
:
ROLLBACK;

1.8

1.8.1

```

Accuracer /
,
SU,
2 ( ) 5
- ,
Multi-Thread. Accuracer
-
,
SU Std SU Pro
SU,
Exclusive TACRDatabase True). (
,
Accuracer
SessionID, TACRQuery TACRTable,
SessionID TACRSession TACRDatabase
TACRQuery, TACRTable
,
Accuracer
1) TACRSession Accuracer :

```

TACRDatabase
 2) TACRQuery TACRDatabase
 TACRTable
 Multi-Thread.

Accuracer

. Accuracer

Accuracer

ACRManager

SQLConsole

Note:

SU

!!!

Exclusive

TACRDatabase

True.

Exclusive

TACRTable

True

1.8.2

Accuracer

Accuracer:

- (Accuracer.)-

- (Accuracer,)-

(,).

(), Accuracer,

SessionID.

Accuracer

()

1

11

(TACRDatabase.Options.MaxSessionCount)

Accuracer

MS Windows

Unix, Linux Novell.

Accuracer

- IS -

- X -

- S -

- IRW -

- RW -

- U -

TACRDataset)

(U-),

TACRDatabase.LockParams.Delay,

TACRDatabase.LockParams.RetryCount,

Lock Mode	X	IS	S	IRW	RW
X	NO	NO	NO	NO	NO
IS	NO	YES	YES	YES	YES
S	NO	YES	YES	YES	NO

IRW	NO	YES	YES	NO	NO
RW	NO	YES	NO	NO	NO

- 1)
- 2)
- 3)

1 2

1.8.3

(ACID):

Atomicity ()

()

Consistency ()

Isolation ()

Durability ()

Accuracer

Accuracer

Accuracer,

Commit () Rollback

Commit

Commit

Rollback

S- () IRW-

SQL- INSERT, UPDATE, DELETE. RW-

Accuracer - READ COMMITTED.

- 1) TACRDatabase - StartTransaction, Commit, Rollback
- 2) SQL - START TRANSACTION, COMMIT, ROLLBACK

(,).

Commit Rollback.

1.9**1.9.1**

```

Accuracer
)
TACRDatabase ( LocalDatabase DatabaseFileName true).
TACRDatabase ( LocalDatabase ConnectionParams false).
UDP.
OpenDatabasesInExclusiveMode true)

```

1)

Accuracer Database Server
Windows.

Windows NT / XP

/install:

```
c:\Accuracer\Utils\Bin\ACRServer\AccuracerDatabaseServer.exe /install
```

/uninstall:

```
c:\Accuracer\Utils\Bin\ACRServer\AccuracerDatabaseServer.exe /uninstall
```

Windows:

```
c:\Accuracer\Utils\Bin\ACRServer\AccuracerDatabaseServer.exe
```

```

True) , ( Active
DatabaseNames DatabaseFileNames.
Demos\Data\DBDemos.adb.
LocalPort TACRServer.
IP-
( LocalHost ). 'localhost'
2)
TACRDatabase
LocalDatabase False,
ConnectionParams (DatabaseName, RemoteHost, RemotePort
LocalPort) (Open Active := True).
:
DatabaseName = DBDemos
RemoteHost = localhost
RemotePort = 6669
LocalPort = 6668
Client
OpenDatabasesInExclusiveMode true,
:
- 5 ;
- 2^31 ;
OpenDatabasesInExclusiveMode false,
:
- 2 ;
- MaxSessionsCount ;
Accuracer Database Server SQL-
:
-
- SQL-
- SQL-
TACRServer.OnSQL
Accuracer Component Reference.

```

Accuracer -

- 1) BDE èèè ODBC
- 2) EasyTable
- 3)

1.10.2 BDE

- 1) DBTransfer
- 2) TTable, TQuery, TSession TDatabase
- TACRTable, TACRQuery, TACRSession TACRDatabase
- 3) DatabaseFileName TACRDatabase
- Alias / Directory.
- 4) Exclusive TACRTable
- TTable
- 5) (TACRDatabase True.),
- Exclusive DBTables uses-
- 6) BDE.

1.10.3 EasyTable

- 1) Convert
- 2) TEasyTable, TEasyQuery, TEasySession TEasyDatabase
- TACRTable, TACRQuery, TACRSession TACRDatabase
- 3) DatabaseFileName TACRDatabase
- 4) Exclusive TACRTable True
- 5) (TACRDatabase True.),
- Exclusive EasyTable Etbl* uses-
- 6) EasyTable.

1.10.4

Accuracer.
 TQuery, TSession TDatabase, TTable,
 ADO, Interbase dbExpress - BDE.
 Accuracer.
 :

1.10.5

1) Accuracer:
 2) DBTransfer Convert
 BDE ODBC (Paradox, Interbase, Access, DBase FoxPro, Oracle,
 SQLServer).

BDE ODBC
 , BDE ODBC,
 DBTransfer,
 < Accuracer>\Utils\Bin\DBTransfer\
 Paradox DBase (FoxPro)
 DBTransfer.
 BDE ODBC, DBTransfer

EasyTable

EasyTable. Convert

MySQL

. MySQLImport.

CSV (comma-separated values)

. CSVImport.


```

1:
TACRTable1.AddIndex('case_ins_index', 'Company', [ixCaseInsensitive]);
TACRTable1.Locate('Company', ['AidAim Software'], [loCaseInsensitive]);

```

```

2:
TACRTable1.AddIndex('complex_index', 'Company;Contact;Phone', []);
TACRTable1.Locate('Company;Contact;Phone', VarArrayOf(['Sight Diver','P']), loPartialKey);

```

```

SELECT
DISTINCT, WHERE, ORDER BY FROM.
ORDER BY
2.
LIKE
: Primary Unique.

```

LockTable / UnlockTable

TACRDataset

```

ACRTable1.LockTable;
try
  ACRTable1.First;
  while not ACRTable1.Eof do
    begin
      // ...
      ACRTable1.Next;
    end;
finally
  ACRTable1.UnockTable;
end;

```

TACRDatabase

TACRTable

True.

Exclusive

: support@aidaim.com.: <http://www.aidaim.com/info/subscr.php>**1.12.2**

Accuracer		:
<u>SQL</u>		<u>TFieldType</u>
Char, FixedChar		ftFixedChar
Varchar, Varchar2, String		ftString
WideChar, FixedWideChar		ftWideString
WideVarchar, WideString	()	ftWideString
Shortint, SignedInt8	8-	ftSmallint
Smallint, SignedInt16	16-	ftSmallint
Integer, SignedInt32	32-	ftInteger
Largeint, Int64, SignedInt64	64-	ftLargeint
Byte, UnsignedInt8		ftWord
Word, UnsignedInt16	16-	ftWord
Cardinal, UnsignedInt32	32-	ftLargeint
AutoInc, AutoincInteger	32-	ftAutoinc
AutoIncShortint		8- ftAutoinc
AutoIncSmallint	16-	ftAutoinc
AutoIncLargeint	64-	ftAutoinc
AutoIncByte		ftAutoinc
AutoIncWord	16-	ftAutoinc
AutoIncCardinal		ftAutoinc

32-

Single			ftFloat
Float, Double			ftFloat
Extended			ftFloat
Boolean, Logical, Bool, Bit			ftBoolean
Currency, Money			ftCurrency
Date			ftDate
Time			ftTime
DateTime	-		ftDateTime
TimeStamp	-		ftTimeStamp
		dbExpress	
Bytes	()	ftBytes
VarBytes	()	ftVarBytes
Blob		Binary Large Object	ftBlob
Graphic			ftGraphic
Memo, Clob			ftMemo
FormattedMemo, FmtMemo			ftFmtMemo
WideMemo, WideClob		Unicode	ftMemo

1.12.3

```

Accuracer
Accuracer
    'Birthday=01/01/1970'.
    DateSeparator, TimeSeparator
    DateToStr/TimeToStr   Filter
    FloatToStr            DecimalSeparator.
    
```

```

IndexName IndexFieldNames
, , , , ,
, /
Accuracer
ftString,
Unicode.
Unicode
Unicode Unicode -
Unicode -
Unicode 256
Unicode ANSI.
Accuracer Unicode ftWideString.
Unicode-

```

```

var ws: WideString;
with MyAccuracer do
begin
//
ws := FieldByName('Unicode').Value;
// do something with data in ws
ws := 'example string';
//
Insert;
FieldByName('Unicode').Value := ws;
Post;
end;

```

```

WideMemo SetWideMemoField
GetWideMemoField TACRDataset:

```

```

var ws: WideString;
with MyAccuracer do
begin
//
ws := 'example string';
//
Insert;
SetWideMemoField(FieldByName('Unicode'), ws);
Post;
//
ws := GetWideMemoField(FieldByName('Unicode'));
end;

```

```

: Windows-
CompareStringW (NT, 2000, XP, 2003). Windows 9x, Me

```

1.12.4

- WideString Delphi 4, C++ Builder 4
- SQLTimeStamp Delphi 4,5, C++ Builder 4,5
- BCD
- SELECT- SQL
- - 2
- - 5
- SU Std
- OnFilterRecord
- RepairTable, GetLastAutoinc SetLastAutoinc

Index

- A -

ABS Function 42
Access 73
ADO 73
Aggregate Functions 29
ALTER TABLE Statement 60
AVG Function 30

- B -

BDE 73
BLOB and Varchar fields 16
BLOB Compression 16
BLOB fields use 15

- C -

CAST Function 49
CEILING Function 44
Client-Server 70
Commit 68
COMMIT Statement 64
Compactness 75
Compression 16
Connections 65
Contents 9
COUNT Function 30
CREATE INDEX Statement 62
CREATE TABLE Statement 56, 58
Creating a table 7
CUMPROD Function 42
CUMSUM Function 42
CURRENT_DATE Function 33
CURRENT_TIME Function 33
CURRENT_TIMESTAMP Function 33

- D -

Data 72, 74
Database 74

Date and Time Functions 32
DAY Function 33
DAYNAME Function 34
DAYOFWEEK Function 34
DBase 73
DELETE Statement 56
Differences from TTable 77
DISTINCT 50
DROP DATABASE Statement 57
DROP INDEX Statement 63
DROP TABLE Statement 62

- E -

EasyTable 73
EXCEPT 50
Exclusive access 65
Export 72, 74
EXTRACT Function 34

- F -

Filtering tables 12
FLOOR Function 43
FoxPro 73
ftFixedChar 16
ftStiring 16
Functions 28

- G -

Getting Help from Technical Support 5
GROUP BY 50
GROUP_CONCAT Function 30

- H -

HEX constants 28
HEX Function 45
HOUR Function 35
How to Buy 6

- I -

Import 72, 74
InMemory tables 6

INSERT Statement 55
 Internationalization and localization 79
 ISNULL Function 40

- J -

JOIN 50

- L -

LASTAUTOINC Function 41
 LENGTH Function 46
 Locks 66
 LOWER Function 46
 LTRIM Function 47

- M -

Mathematical Functions 41
 MAX Function 31
 Migration 72
 MIN Function 31
 MINUS 50
 MINUTE Function 35
 Miscellaneous Functions 40
 MOD Function 43
 MONTH Function 36
 MONTHNAME Function 36
 Moving 74
 Moving Data 72
 MSECOND Function 36
 Multi-Thread 65
 Multi-User 65

- N -

Naming conventions 20
 Navigating Tables 9
 Network 70
 NOW Function 33

- O -

ODBC 73
 Operators 26
 ORDER BY 50

Other Functions 40
 Overview 19

- P -

Paradox 73
 Parameters 29
 Performance 75
 POS Function 47
 POWER Function 45

- Q -

QUARTER Function 36

- R -

RAND Function 45
 RANDOM Function 45
 Record locks 66
 Rollback 68
 ROLLBACK Statement 64
 ROUND Function 44
 RTRIM Function 47

- S -

SECOND Function 37
 SELECT Statement 50
 Sessions 65
 Setting up a table component 6
 SIGN Function 43
 Sorting records 14
 Speed 75
 SQL 19
 SQL Functions 28
 Start transaction 68
 START TRANSACTION Statement 63
 String Functions 46
 SUBSTRING Function 48
 SUM Function 32
 Supported data types 78
 SYSDATE Function 33

- T -

Table locks 66
Tables 74
Third Party DB systems 74
TOBLOB Function 49
TODATE Function 37
TOP 50
TOSTRING Function 38
Transactions 68
TRIM Function 48
TRUNCATE Function 44
Tuning and Optimizations 75
Type Conversion Functions 48

- U -

Unicode 79
UNION 50
UPDATE Statement 55
UPPER Function 48
Using parameters 29

- V -

Varchar 16
Varchar Compression 16

- W -

WEEKDAY Function 40
WideMemo 79
WideString 79

- Y -

YEAR Function 40

- Z -

2

2

41

17

28

Endnotes 2... (after index)

Back Cover