BDE API Overview

Available BDE 4.0 engine functions by type:

1. Database functions

Each function listed below returns information about a specific database, available databases, or performs a database-related task, such as opening or closing a database.

DbiCloseDatabase	Closes a database and all tables associated with this database handle.
DbiGetDatabaseDesc	Retrieves the description of the specified database from the configuration file.
DbiGetDirectory	Retrieves the current working directory or the default directory.
DbiOpenDatabase	Opens a database in the current session and returns a database handle.
DbiOpenDatabaseList	Creates an in-memory table containing a list of accessible databases and their descriptions.
DbiOpenFileList	Opens a cursor on the virtual table containing all the tables accessible by the client application and their descriptions.
DbiOpenIndexList	Opens a cursor on an in-memory table listing the indexes on a specified table, a-long with their descriptions.
DbiOpenTableList	Creates an in-memory table with information about all the tables accessible to the client application.

2. Environment and configuration functions

Each function listed below returns information about the client application environment, such as the supported table, field and index types for the driver type, or the available driver types. Functions in this category can also perform tasks that affect the client application environment, such as loading a driver.

DbiAddAlias	Adds an alias to the BDE configuration file (IDAPI.CFG).
DbiAddDriver	Adds a driver to the BDE configuration file (IDAPI.CFG). NEW FUNCTION BDE 4.0
DbiAnsiToNative	Multipurpose translate function.
DbiDebugLayerOptions	Activates, deactivates, or sets options for the BDE debug layer. OBSOLETE FUNCTION BDE 4.0
DbiDeleteAlias	Deletes an alias from the BDE configuration file (IDAPI.CFG).
DbiDeleteDriver	Deletes a driver from the BDE configuration file (IDAPI.CFG). NEW FUNCTION BDE 4.0
DbiDllExit	Prepares the BDE to be disconnected within a DLL. NEW FUNCTION BDE 4.0
DbiExit	Disconnects the client application from BDE.
DbiGetClientInfo	Retrieves system-level information about the client application environment.
DbiGetDriverDesc	Retrieves a description of a driver.
DbiGetLdName	Retrieves the name of the language driver associated with the specified object name (table name).
DbiGetLdObj	Retrieves the language driver object associated with the given cursor.

DbiGetNetUserName	Retrieves the user's network login name. User names should be available for all networks supported by Microsoft Windows.
DbiGetProp	Returns a property of an object.
DbiGetSysConfig	Retrieves BDE system configuration information.
DbiGetSysInfo	Retrieves system status and information.
DbiGetSysVersion	Retrieves the system version information, including the BDE version number, date, and time, and the client interface version number.
DbiInit	Initializes the BDE environment.
DbiLoadDriver	Load a given driver.
DbiNativeToAnsi	Translates a string in the native language driver to an ANSI string.
DbiOpenCfgInfoList	Returns a handle to an in-memory table listing all the nodes in the configuration file accessible by the specified path.
DbiOpenDriverList	Creates an in-memory table containing a list of driver names available to the client application.
DbiOpenFieldTypesList	Creates an in-memory table containing a list of field types supported by the table type for the driver type.
DbiOpenFunctionArgList	Returns a list of arguments to a data source function.
DbiOpenFunctionList	Returns a description of a data source function.
DbiOpenIndexTypesList	Creates an in-memory table containing a list of all supported index types for the driver type.
DbiOpenLdList	Creates an in-memory table containing a list of available language drivers.
DbiOpenTableList	Creates an in-memory table with information about all the tables accessible to the client application.
DbiOpenTableTypesList	Creates an in-memory table listing table type names for the given driver.
DbiOpenUserList	Creates an in-memory table containing a list of users sharing the same network file.
DbiSetProp	Sets the specified property of an object to a given value.
DbiUseIdleTime	Allows BDE to accomplish background tasks during times when the client application is idle. OBSOLETE FUNCTION BDE 4.0

3. Session functions

Each function listed below returns information about a session or performs a task that affects the session, such as starting a session or adding a password.

DbiAddPassword	Adds a password to the current session.
DbiCheckRefresh	Checks for remote updates to tables for all cursors in the current session, and refreshes the cursors if changed.
DbiCloseSession	Closes the session associated with the given session handle.
DbiDropPassword	Removes a password from the current session.
DbiGetCallBack	Returns a pointer to the function previously registered by the client for the given callback type.
DbiGetCurrSession	Returns the handle associated with the current session.
DbiGetDateFormat	Gets the date format for the current session.

DbiGetNumberFormat	Gets the number format for the current session.
DbiGetSesInfo	Retrieves the environment settings for the current session.
DbiGetTimeFormat	Gets the time format for the current session.
DbiRegisterCallBack	Registers a callback function for the client application.
DbiSetCurrSession	Sets the current session of the client application to the session associated with hSes.
DbiSetDateFormat	Sets the date format for the current session.
DbiSetNumberFormat	Sets the number format for the current session.
DbiSetPrivateDir	Sets the private directory for the current session.
DbiSetTimeFormat	Sets the time format for the current session.
DbiStartSession	Starts a new session for the client application.

4. Error-handling functions

Each function listed below returns error handling information or performs a task that relates to error handling.

DbiGetErrorContext	After receiving an error code back from a call, enables the client to probe BDE for more specific error information.
DbiGetErrorEntry	Returns the error description of a specified error stack entry.
DbiGetErrorInfo	Provides descriptive error information about the last error that occurred.
DbiGetErrorString	Returns the message associated with a given error code.

5. Lock functions

Each function listed below returns information about lock status or acquires or releases a lock at the table or record level.

DbiAcqPersistTableLock	Acquires an exclusive persistent lock on the table preventing other users from using the table or creating a table of the same name.
DbiAcqTableLock	Acquires a table-level lock on the table associated with the given cursor.
DbiGetRecord	Record positioning functions have a lock parameter.
DbiIsRecordLocked	Checks the lock status of the current record.
DbiIsTableLocked	Returns the number of locks of a specified type acquired on the table associated with the given session.
DbiIsTableShared	Determines whether the table is physically shared or not.
DbiOpenLockList	Creates an in-memory table containing a list of locks acquired on the table.
DbiOpenUserList	Creates an in-memory table containing a list of users sharing the same network file.
DbiRelPersistTableLock	Releases the persistent table lock on the specified table.
DbiRelRecordLock	Releases the record lock on either the current record of the cursor or only the locks acquired in the current session.
DbiRelTableLock	Releases table locks of the specified type associated with the current session (the session in which the cursor was created).
DbiSetLockRetry	Sets the table and record lock retry time for the current session.

6. Cursor functions

Each function listed below returns information about a cursor, or performs a task that performs a cursor-related task such as positioning of a cursor, linking of cursors, creating and closing cursors, counting of records associated with a cursor, filtering, setting and comparing bookmarks, and refreshing all buffers associated with a cursor.

DbiActivateFilter	Activates a filter.
DbiAddFilter	Adds a filter to a table, but does not activate the filter (the record set is not yet altered).
DbiApplyDelayedUpdates	When cached updates cursor layer is active, writes all modifications made to cached data to the underlying database.
DbiBeginDelayedUpdates	Creates a cached updates cursor layer so that users can make extended changes to temporarily cached table data without writing to the actual table, thereby minimizing resource locking.
DbiBeginLinkMode	Converts a cursor to a link cursor. Given an open cursor, prepares for linked access. Returns a new cursor.
DbiCloneCursor	Creates a new cursor (clone cursor) which has the same result set as the given cursor (source cursor).
DbiCloseCursor	Closes a previously opened cursor.
DbiCompareBookMarks	Compares the relative positions of two bookmarks in the result set associated with the cursor.
DbiDeactivateFilter	Temporarily stops the specified filter from affecting the record set by turning the filter off.
DbiDropFilter	Deactivates and removes a filter from memory, and frees all resources.
DbiEndDelayedUpdates	Closes a cached updates cursor layer ending the cached updates mode.
DbiEndLinkMode	Ends linked cursor mode, and returns the original cursor.
DbiExtractKey	Retrieves the key value for the current record of the given cursor or from the supplied record buffer.
DbiForceRecordReread	Rereads a single record from the server on demand, refreshing one row only, rather than clearing the cache.
DbiForceReread	Refreshes all buffers associated with the cursor, if necessary.
DbiFormFullName	Returns the fully qualified table name.
DbiGetBookMark	Saves the current position of a cursor to the client-supplied buffer called a bookmark.
DbiGetCursorForTable	Finds the cursor for the given table.
DbiGetCursorProps	Returns the properties of the cursor.
DbiGetExactRecordCount	Retrieves the current exact number of records associated with the cursor. NEW FUNCTION BDE 4.0
DbiGetFieldDescs	Retrieves a list of descriptors for all the fields in the table associated with the cursor.
DbiGetLinkStatus	Returns the link status of the cursor.
DbiGetNextRecord	Retrieves the next record in the table associated with the cursor.
DbiGetPriorRecord	Retrieves the previous record in the table associated with the given cursor.
DbiGetProp	Returns a property of an object.

DbiGetRecord	Retrieves the current record, if any, in the table associated with the cursor.
DbiGetRecordCount	Retrieves the current number of records associated with the cursor.
DbiGetRecordForKey	Finds and retrieves a record matching a key and positions the cursor on that record.
DbiGetRelativeRecord	Positions the cursor on a record in the table relative to the current position of the cursor.
DbiGetSeqNo	Retrieves the sequence number of the current record in the table associated with the cursor.
DbiLinkDetail	Establishes a link between two tables such that the detail table has its record set limited to the set of records matching the linking key values of the master table cursor.
DbiLinkDetailToExp	Links the detail cursor to the master cursor using an expression.
DbiMakePermanent	Changes a temporary table created by DbiCreateTempTable into a permanent table.
DbiOpenTable	Opens the given table for access and associates a cursor handle with the opened table.
DbiResetRange	Removes the specified table's limited range previously established by the function DbiSetRange.
DbiSaveChanges	Forces all updated records associated with the cursor to disk.
DbiSetFieldMap	Sets a field map of the table associated with the given cursor.
DbiSetProp	Sets the specified property of an object to a given value.
DbiSetRange	Sets a range on the result set associated with the cursor.
DbiSetToBegin	Positions the cursor to BOF (just before the first record).
DbiSetToBookMark	Positions the cursor to the location saved in the specified bookmark.
DbiSetToCursor	Sets the position of one cursor (the destination cursor) to that of another (the source cursor).
DbiSetToEnd	Positions the cursor to EOF (just after the last record).
DbiSetToKey	Positions an index-based cursor on a key value.
DbiSetToRecordNo	Positions the cursor of a dBASE table to the given physical record number.
DbiSetToSeqNo	Positions the cursor to the specified sequence number of a Paradox table.
DbiUnlinkDetail	Removes a link between two cursors.

7. Index functions

Each function listed below returns information about an index or indexes, or performs a task that affects an index, such as dropping it, deleting it, or adding it.

DbiAddIndex	Creates an index on an existing table.
DbiCloseIndex	Closes the specified index on a cursor.
DbiCompareKeys	Compares two key values based on the current index of the cursor.
DbiDeleteIndex	Drops an index on a table.
DbiExtractKey	Retrieves the key value for the current record of the given cursor or from the supplied record buffer.
DbiGetIndexDesc	Retrieves the properties of the given index associated with the cursor.

DbiGetIndexDescs	Retrieves index properties.
DbiGetIndexForField	Returns the description of any useful index on the specified field.
DbiGetIndexSeqNo	Retrieves the ordinal number of the index in the index list of the specified cursor.
DbiGetIndexTypeDesc	Retrieves a description of the index type.
DbiOpenIndex	Opens the index for the table associated with the cursor.
DbiRegenIndex	Regenerates an index to make sure that it is up-to-date (all records currently in the table are included in the index and are in the index order).
DbiRegenIndexes	Regenerates all out-of-date indexes on a given table.
DbiSwitchToIndex	Allows the user to change the active index order of the given cursor.

8. Query functions

Each function listed below performs a query task, such as preparing and executing a SQL or QBE query.

DbiGetProp	Returns a property of an object.
DbiQAlloc	Allocates a new statement handle for a prepared query.
DbiQExec	Executes the previously prepared query identified by the supplied statement handle and returns a cursor to the result set, if one is generated.
DbiQExecDirect	Executes a SQL or QBE query and returns a cursor to the result set, if one is generated.
DbiQExecProcDirect	Executes a stored procedure and returns a cursor to the result set, if one is generated.
DbiQFree	Frees the resources associated with a previously prepared query identified by the supplied statement handle.
DbiQGetBaseDescs	Returns the original database, table, and field names of the fields that make up the result set of a query.
DbiQInstantiateAnswer	Creates a permanent table from the cursor to the result set.
DbiQPrepare	Prepares a SQL or QBE query for execution, and returns a handle to a statement containing the prepared query.
DbiQPrepareProc	Prepares and optionally binds parameters for a stored procedure.
DbiQSetParams	Associates data with parameter markers embedded within a prepared query.
DbiQSetProcParams	Binds parameters for a stored procedure prepared with DbiQPrepareProc.
DbiSetProp	Sets the specified property of an object to a given value.
DbiValidateProp	Validates a property.

9. Table functions

Each function listed below returns information about a specific table, such as all the locks acquired on the table, all the referential integrity links on the table, the indexes open on the table, or whether or not the table is shared. Functions in this category can also perform a table-wide operation, such as copying and deleting.

DbiBatchMove	Appends, updates, subtracts, and copies records or fields from a source table to a destination table.
DbiCopyTable	Duplicates the specified source table to a destination table.
DbiCreateInMemTable	Creates a temporary, in-memory table.

DbiCreateTable	Creates a table.
DbiCreateTempTable	Creates a temporary table that is deleted when the cursor is closed, unless the call is followed by a call to DbiMakePermanent.
DbiDeleteTable	Deletes a table.
DbiDoRestructure	Changes the properties of a table.
DbiEmptyTable	Deletes all records from the table associated with the specified table cursor handle or table name.
DbiGetTableOpenCount	Returns the total number of cursors that are open on the specified table.
DbiGetTableTypeDesc	Returns a description of the capabilities of the table type for the driver type.
DbiIsTableLocked	Returns the number of locks of a specified type acquired on the table associated with the given session.
DbiIsTableShared	Determines whether the table is physically shared or not.
DbiMakePermanent	Changes a temporary table created by DbiCreateTempTable into a permanent table.
DbiOpenFamilyList	Creates an in-memory table listing the family members associated with a specified table.
DbiOpenFieldList	Creates an in-memory table listing the fields in a specified table and their descriptions.
DbiOpenIndexList	Opens a cursor on an in-memory table listing the indexes on a specified table, along with their descriptions.
DbiOpenLockList	Creates an in-memory table containing a list of locks acquired on the table associated with the cursor.
DbiOpenRintList	Creates an in-memory table listing the referential integrity links for a specified table, along with their descriptions.
DbiOpenSecurityList	Creates an in-memory table listing record-level security information about a specified table.
DbiOpenTable	Opens the given table for access and associates a cursor handle with the opened table.
DbiPackTable	Optimizes table space by rebuilding the table associated with the cursor and releasing any free space.
DbiQInstantiateAnswer	Creates a permanent table from a cursor handle.
DbiRegenIndexes	Regenerates all out-of-date indexes on a given table.
DbiRenameTable	Renames the table and all of its resources to the new name specified.
DbiSaveChanges	Forces all updated records associated with the table to disk.
DbiSortTable	Sorts an opened or closed table, either into itself or into a destination table. There are options to remove duplicates, to enable case-insensitive sorts and special sort functions, and to control the number of records sorted.

10. Data access functions

Each function listed below accesses data in a table, such as retrieving data from a specified BLOB field or from the record buffer.

DbiAppendRecord	Appends a record to the end of the table associated with the given cursor.
DbiDeleteRecord	Deletes the current record of the given cursor.

DbiFreeBlob	Closes the BLOB handle located within the specified record buffer.
DbiGetBlob	Retrieves data from the specified BLOB field.
DbiGetBlobHeading	Retrieves information about a BLOB field from the BLOB heading in the record buffer.
DbiGetBlobSize	Retrieves the size of the specified BLOB field in bytes.
DbiGetField	Retrieves the data contents of the requested field from the record buffer.
DbiGetFieldDescs	Retrieves a list of descriptors for all the fields in the table associated with the cursor.
DbiGetFieldTypeDesc	Retrieves a description of the specified field type.
DbiInitRecord	Initializes the record buffer to a blank record according to the data types of the fields.
DbiInsertRecord	Inserts a new record into the table associated with the given cursor.
DbiModifyRecord	Modifies the current record of table associated with the cursor with the data supplied.
DbiOpenBlob	Prepares the cursor's record buffer to access a BLOB field.
DbiPutBlob	Writes data into an open BLOB field.
DbiPutField	Writes the field value to the correct location in the supplied record buffer.
DbiReadBlock	Reads a specified number of records (starting from the next position of the cursor) into a buffer.
DbiSaveChanges	Forces all updated records associated with the cursor to disk.
DbiSetFieldMap	Sets a field map of the table associated with the given cursor.
DbiTruncateBlob	Shortens the size of the contents of a BLOB field, or deletes the contents of a BLOB field from the record, by shortening it to zero.
DbiUndeleteRecord	Undeletes a dBASE record that has been marked for deletion (a "soft" delete).
DbiVerifyField	Verifies that the data specified is a valid data type for the field specified, and that all validity checks in place for the field are satisfied. It can also be used to check if a field is blank.
DbiWriteBlock	Writes a block of records to the table associated with the cursor.

11. Transaction functions

Each function listed below begins, ends, or inquires about the status of a transaction.

DbiBeginTran	Begins a transaction.
DbiEndTran	Ends a transaction.
DbiGetTranInfo	Retrieves the transaction state.

12. Capability or schema functions

Each function listed below returns information about capabilities or the schema.

DbiOpenCfgInfoList	Returns a handle to an in-memory table listing all the nodes in the configuration file accessible by the specified path.
DbiOpenDatabaseList	Creates an in-memory table containing a list of accessible databases and their descriptions.

DbiOpenDriverList	Creates an in-memory table containing a list of driver names available to the client application.
DbiOpenFamilyList	Creates an in-memory table listing the family members associated with a specified table.
DbiOpenFieldList	Creates an in-memory table listing the fields in a specified table and their descriptions.
DbiOpenFieldTypesList	Creates an in-memory table containing a list of field types supported by the table type for the driver type.
DbiOpenFunctionArgList	Returns a list of arguments to a data source function.
DbiOpenFunctionList	Returns a description of a data source function.
DbiOpenIndexList	Opens a cursor on an in-memory table listing the indexes on a specified table, along with their descriptions.
DbiOpenIndexTypesList	Creates an in-memory table containing a list of all supported index types for the driver type.
DbiOpenLockList	Creates an in-memory table containing a list of locks acquired on the table.
DbiOpenRintList	Creates an in-memory table listing the referential integrity links for a specified table, along with their descriptions.
DbiOpenSecurityList	Creates an in-memory table listing record-level security information about a specified table.
DbiOpenTableList	Creates an in-memory table with information about all the tables accessible to the client application.
DbiOpenTableTypesList	Creates an in-memory table listing table type names for the given driver.
DbiOpenVchkList	Creates an in-memory table containing records with information about validity checks for fields within the specified table.

13. Date/time/number format functions

Each function listed below sets or retrieves date or time, or decodes/encodes date and time into or from a timestamp.

DbiBcdFromFloat	Converts FLOAT data to binary coded decimal (BCD) format.
DbiBcdToFloat	Converts binary coded decimal (BCD) data to FLOAT format.
DbiDateDecode	ecodes DBIDATE into separate month, day and year components.
DbiDateEncode	Encodes separate date components into date for use by DbiPutField and other functions.
DbiGetDateFormat	Gets the date format for the current session.
DbiGetNumberFormat	Gets the number format for the current session.
DbiGetTimeFormat	Gets the time format for the current session.
DbiSetDateFormat	Sets the date format for the current session.
DbiSetNumberFormat	Sets the number format for the current session.
DbiSetTimeFormat	Sets the time format for the current session.
DbiTimeDecode	Decodes time into separate components (hours, minutes, milliseconds).
DbiTimeEncode	Encodes separate time components into time for use by DbiPutField and other functions.

DbiTimeStampDecode	Extracts separate encoded date and time components from the timestamp.
DbiTimeStampEncode	Encodes the encoded date and encoded time into a timestamp.

Original resource:	The Delphi Pool
Author:	Unknown
Added:	2013-01-28
Last updated:	2013-01-28