Type Compatibility

The following tables shows which pairs of types will be considered compatible when comparing values of different type:

	Bool	Byte	DateTime	Decimal	Double	Int16	Int32	Int64	Single	String (**)	Blob	Clob (**)
Bool (*)	yes	partial	no	partial	partial	partial	partial	partial	partial	partial	no	partial
Byte		yes	no	yes	yes	yes	yes	yes	yes	partial	no	partial
DateTime			yes	no	no	no	no	no	no	partial	no	partial
Decimal				yes	yes	yes	yes	yes	yes	partial	no	partial
Double					yes	yes	yes	yes	yes	partial	no	partial
Int16						yes	yes	yes	yes	partial	no	partial
Int32							yes	yes	yes	partial	no	partial
Int64								yes	yes	partial	no	partial
Single									yes	partial	no	partial
String										yes	no	yes
Blob											yes	no
Clob												yes

(*) When comparing a Boolean to a numeric, they will be compatible when the numeric value is 0 or 1.The values are considered equal when the Boolean is true and the numeric is 1, or the Boolean is false and the numeric is 0. For all other cases, the Compare function will return FdoCompareType_Undefined.

(**) For the "partial" cases, compatibility depends on the value of the String or Clob. When compared with values of other types, the two values are considered compatible if the String or Clob value can be converted to the other type. Otherwise the values will be considered incompatible. Please refer to the **Fdo DataValue Type Conversion** document for information on how types will be converted.

Type Conversions

Different data types have different formats and collations (ordering sequences) for their possible values. When comparing values of different types, it is necessary to use a common format and use a common collation.. When comparing pairs of values of these types FdoDataValue::Compare() will behave as if both values were converted to the common type before comparing. However, the FdoDataValue::Compare() implementation may not necessarily carry out all of these type conversions; it will just behave as if it did.

The following table indicates the common type that will determine the format and collation to be used for each pair of compatible types. However, it is applicable only when the values being compared are compatible (See Section 3 table for compatibility info).

	Bool	Byte	Date Time	Decimal	Double	Int16	Int32	Int64	Single	String (**)	Blob	Clob (**)
Bool	Bool	Bool		Bool	Bool	Bool	Bool	Bool	Bool	Bool		Bool
Byte		Byte		Decimal	Double	Int16	Int32	Int64	Single	Byte		Byte
Date Time			Date Time							Date Time		Date Time
Decimal				Decimal	Double	Decimal	Decimal	(*)	Decimal	Decimal		Decimal
Double					Double	Double	Double	(*)	Double	Double		Double
Int16						Int16	Int32	Int64	Single	Int16		Int16
Int32							Int32	Int64	(*)	Int32		Int32
Int64								Int64	(*)	Int64		Int64
Single									Single	Single		Single
String										String		Clob
Blob											Blob	
Clob												Clob

(*) There actually is no common type with large enough value range and precision to handle these pairs of types. However, these types are all numeric so it is sufficient to say that FdoDataValue::Compare() will do a numeric comparison on them.

(**) Note that numeric ordering will be used to compare Strings against values of numeric type.