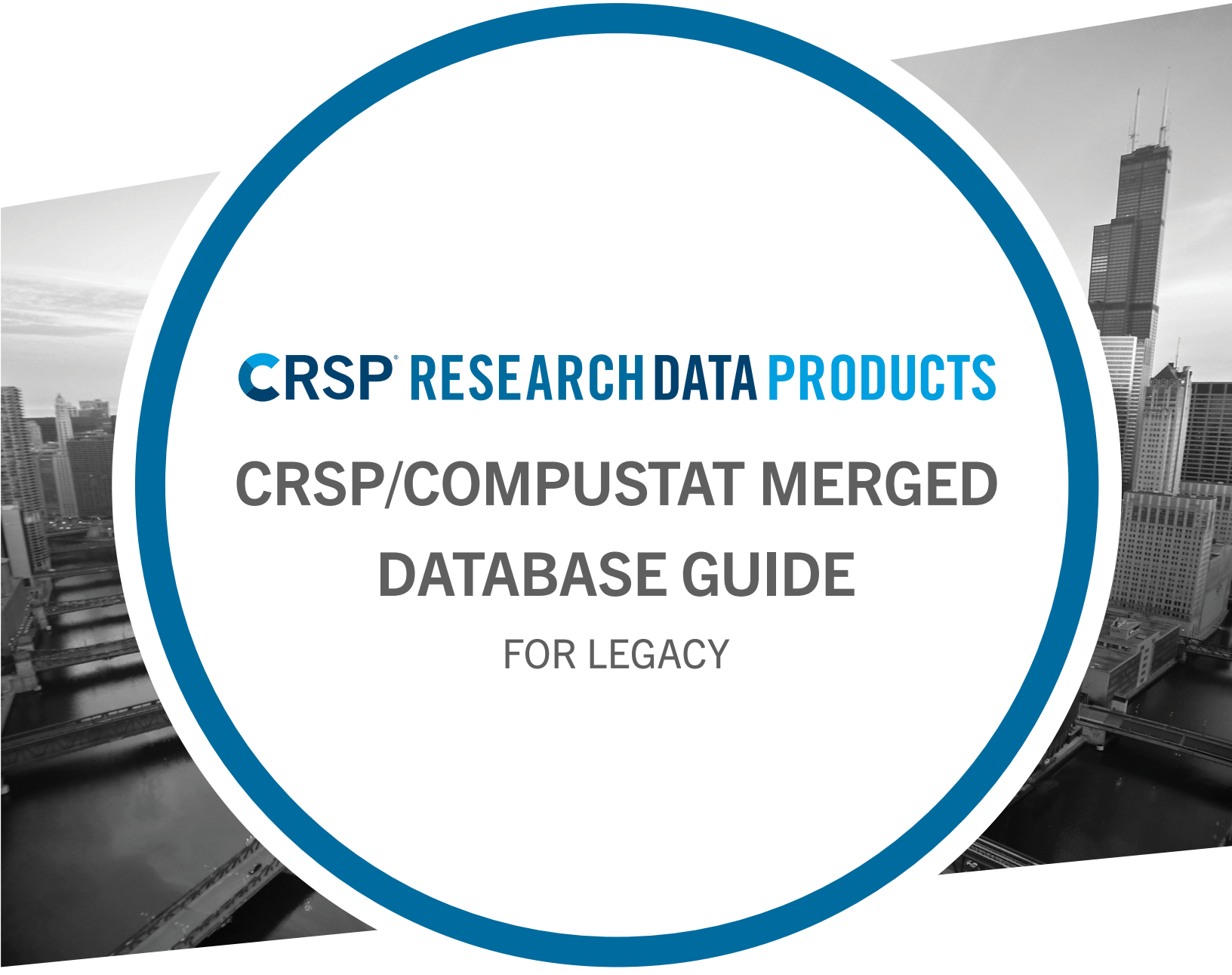

CRSP® RESEARCH DATA PRODUCTS

CRSP/COMPUSTAT MERGED DATABASE GUIDE

FOR LEGACY

CRSP® CENTER FOR RESEARCH IN SECURITY PRICES
An Affiliate of the University of Chicago

500 W Madison St Ste 2900 • Chicago, IL 60661

Phone: 312-263-6400 • Email: support@crsp.org

Learn more at crsp.org

Table of Contents

CHAPTER 1: DATA DEFINITIONS

This section describes the CRSP Link and Compustat variables and structures supported by the CRSP/Compustat Merged database. For complete definitions, codes, and formulas for Compustat items, see the documentation and resources provided to you by Compustat.

DATA ORGANIZATION

Compustat data are organized by company and security around Compustat's Permanent SPC Identifier, GVKEY, and issue identifier, IID. Secondary identifiers are available in the header and link history that can be used to cross-reference companies to GVKEYs.

A defined structure for Compustat data is used to store all available Compustat and CRSP Link data for a GVKEY. Each structure is broken down into items. All structures are built from three basic data categories: headers, event data arrays, and time series.

- Headers have no time component. They are a collection of data items with one instance for each gvkey. Examples of header data items are current identifiers and date ranges.
- Event data arrays are collections of records, each describing a change in status or a new event. All data items describing the event type are included in each record. These always include one or more data items that describe the effective date range or the effective date of the event. A count of the number of events being referenced is available for each event data array.
- Time-series is a collection of records tied to a specific calendar of time periods. Each time-series has a beginning and ending period and exactly one record of information for each period in that range. A time-series record can include one or more data items describing the period.

The data items defined within each data category are determined by the available Compustat or CRSP Link data for that data type.

DATA ITEMS

Data definitions include data items provided by Compustat as well as structures and supplementary data items provided by CRSP. All data items include a mnemonic and field name. With the exception of roughly a dozen data items, the mnemonics of Compustat data items used in the CCM database match the name provided by Compustat. No further definitions for Compustat data items are provided in this guide except clarification on mnemonics and usage for a data item that may be used differently by Compustat in different files. Go to www.compustatresources.com/support/index.html. Supplementary CRSP data items include complete definitions.

Implicit in every structure is CCMID, which may be PERMNO, GVKEY, or GVKEYX, depending upon what identifier key is needed for data access. GVKEY is a unique permanent number assigned by Compustat, that can be used to identify a Compustat record in different updates if name or other identifying information changes. GVKEY is the primary key in the CRSP/Compustat Merged Database. Data are sorted and organized by this field.

ITEM OVERVIEW – ITM_NAMES

Each Compustat item in the CCM database has a unique mnemonic text name, itm_name, maintained by CRSP. The CRSP item names match the Compustat mnemonic names wherever possible. In some rare instances, CRSP must provide a different name from Compustat's in order to maintain uniqueness across the Compustat data groups and all CRSP products supported by CRSPAccess.

The following table is a comprehensive list of cases where the CRSP itm_name used does not match Compustat's mnemonic.

Compustat mnemonic	CRSP itm_name	Description	Definition
BETA	XPFBETA	Data item	Beta
DVPSXM	XDVPXSM	Data item	Index Monthly Dividend
PRC	XPFPRC	Data item	Participation Rights Certificates
PRCCM	XPRCCM	Data item	Index Price – Close Monthly
PRCHM	XPRCHM	Data item	Index Price – High Monthly
PRCLM	XPRCLM	Data item	Index Price – Low Monthly
PRC_DC	XPFPRC_DC	Data code	Participation Rights Certificates Data Code
PRC_FN	XPFPRC_FN	Footnote	Participation Rights Certificates Footnote
RET	XPFRET	Data item	Total RE Property
RET_DC	XPFRET_DC	Data code	Total RE Property Data Code
RET_FN	XPFRET_FN	Footnote	Total RE Property Footnote
YEAR	YEARQ	Data item	Year Quarterly

MASTER, HEADER, HEADER HISTORY, AND LINK HISTORY DATA

Descriptive structures include the Master, Company and Security Header and Header History, and Link History Data.

MASTER DEFINED STRUCTURE

The master structure contains CCM Company identification and range data.

Mnemonic	Field Name	Format
BEGQTR	Quarterly date of earliest data (yyyy.q)	integer
BEGYR	Annual date of earlist data (yyyymmdd)	integer
CBEGDT	First date of Compustat data	integer
CCMID	Permanent record identifier for Compustat company or index data, represents GVKEY for company, GVKEYX for index	integer
CCMIDTYPE	Type of key for Compustat data. 1 = company data, 2 = index data	integer
CENDT	Last date of Compustat data	integer
ENDQTR	Quarterly date of last data (yyyy.q)	integer
ENDYR	Annual date of last data (yyyymmdd)	integer

COMPANY DEFINED STRUCTURE

The company structure contains CCM Company Header information.

Mnemonic	Field Name	Format
ADD1-4	Address lines 1-4	character
ADDZIP	Postal code	character
BUSDESC	Business description	character
CIK	CIK number	character
CITY	City	character
CONM	Company name	character
CONML	Company legal name	character
COSTAT	Postal code	character
COUNTY	County code	character
DLDTE	Research company deletion date	integer

Mnemonic	Field Name	Format
DLRSN	Research company reason for deletion	character
EIN	Employer identification number	character
FAX	Fax number	character
FIC	ISO Country code of incorporation	character
FYRC	Fiscal year end (current)	integer
GGROUP	GICS groups	character
GIND	GICS industries	character
GSECTOR	GICS sectors	character
GSUBIND	GICS sub-industries	character
IDBFLAG	International/Domestic/ Both indicator	character
INCORP	State/Province of incorporation code	character
IPODATE	Company initial public offering date	integer
LOC	ISOCountry code/ headquarters	character
NAICS	North American Industry Classification Code	character
PHONE	Phone number	character
PRICAN	Primary Issue Tag - Canada	character
PRIROW	Primary Issue Tag – rest of world	character
PRIUSA	Primary Issue Tag - USA	character
SIC	SIC code	integer
SPCINDCD	S&P industry sector code - reference	integer
SPCSECCD	S&P economic sector code - reference	integer
STATE	State/Province	character
STKO	Stock ownership code	integer
WEBURL	Website address	character

IDX_INDEX DEFINED STRUCTURE

IDX_Index structure contains index header information.

Mnemonic	Field Name	Format
IDX13KEY	13 character key	character
IDXCSTFLG	Index constituent flag	character
INDEXCAT	Index category code	character
INDEXGEO	Index geographical area	character
INDEXTYPE	Index type	character
INDEXVAL	Index value	character
SPII	S&P industry index code	integer
SPMI	S&P major index code	integer
TICI	Issue trading ticker	character
XCONM	Company Name (Index)	character
XINDEXID	Index ID	character
XTIC	Ticker/trading symbol (index)	character

SPIND DEFINED STRUCTURE

The SPIND structure contains pre-GICS S&P Index header information.

Mnemonic	Field Name	Format
SPIID	S&P Industry ID	integer
SPIMID	S&P Major Index ID	integer
SPITIC	S&P Index ticker	character
SPIDESC	S&P Index industry description/reference	character

COMPHIST DEFINED STRUCTURE

The COMPHIST structure contains Compustat Company Header history.

Mnemonic	Field Name	Format
HCHGDT	Comphist description effective date	integer
HCHGENDDT	Comphist description last effective date	integer
HDLDTE	Historical research company – deletion date	integer
HFYRC	Historical fiscal year end month / current	integer
HIPODATE	Historical company official public offering date	integer
HSIC	Historical SIC Code	integer
HSPCINDCD	Historical S&P Industry code	integer
HSPCSECCD	Historical S&P Economic sector code	integer
HSTKO	Historical stock ownership code	integer
HADD1…4	Historical address lines 1-4	character
HADDZIP	Historical postal code	character
HBUSDESC	Historical business description	character
HCIK	Historical CIK number	character
HCITY	Historical city	character
HCONM	Historical company name	character
HCONML	Historical legal company name	character
HCOSTAT	Historical active/inactive status marker	character
HCOUNTY	Historical county code	character
HDLRSN	Historical research company reason for deletion	character
HEIN	Historical employer identification number	character
HFAX	Historical fax number	character
HFIC	Historical ISO country code / incorporation	character
HGGROUP	Historical GICS group	character
HGIND	Historical GICS industries	character
HGSECTOR	Historical GICS sector	character
HGSUBIND	Historical GICS sub-industries	character
HIDBFLAG	Historical international, domestic, both indicator	character
HINCORP	Historical state/province of incorporation code	character
HLOC	Historic ISO country code/ headquarters	character
HNAICS	Historical NAICS codes	character
HPHONE	Historical phone number	character
HPRICAN	Historical primary issue tag - Cananda	character
HPRIROW	Historical primary issue tag – rest of world	character

Mnemonic	Field Name	Format
HPRIUSA	Historical primary issue tag - US	character
HSTATE	Historical state/province	character
HWEBURL	Historical website url	character

CSTHIST DEFINED STRUCTURE

The CSTHIST structure contains the header history from the legacy CRSP/Compustat Merged database that was created from Compustat FTP files.

Mnemonic	Field Name	Format
CST_CHGDT	CST History effective date	integer
CST_CHGENDDT	CST History last effective date	integer
CST_DNUM	CST History industry code	integer
CST_FILE	CST History file identification code	integer
CST_ZLIST	CST History exchange listing and S&P Index code	integer
CST_STATE	CST History state identification code	integer
CST_COUNTY	CST History county identification code	integer
CST_STINC	CST History state incorporation code	integer
CST_FINC	CST History foreign incorporation code	integer
CST_XREL	CST History industry index relative code	integer
CST_STK	CST History stock ownership code	integer
CST_DUP	CST History duplicate file code	integer
CST_CCNDX	CST History current Canadian index code	integer
CST_GICS	CST History Global Industry Classification Standard Code	integer
CST_IPODT	CST History IPO date	integer
CST_FUNDF1	CST History fundamental file identification code 1	integer
CST_FUNDF2	CST History fundamental file identification code 2	integer
CST_FUNDF3	CST History fundamental file identification code 3	integer
CST_NAICS	CST History North American Industry Classification	character
CST_CPSPIN	CST History primary S&P Index marker	character
CST_CSSPIN	CST History subset S&P Index marker	character
CST_CSSPII	CST History secondary S&P Index marker	character
CST_SUBDBT	CST History current S&P subordinated debt rating	character
CST_CPAPER	CST History current S&P commercial paper rating	character
CST_SDBT	CST History current S&P senior debt rating	character
CST_SDBTIM	CST History current S&P senior debt rating - footnote	character
CST_CNUM	CST History CUSIP issuer code	character
CST_CIC	CST History issuer number	character
CST_CONAME	CST History company name	character
CST_INAME	CST History industry name	character
CST_SMBL	CST History stock ticker symbol	character
CST_EIN	CST History employer identification number	character
CST_INCORP	CST History incorporation ISO country code	character

LINK DEFINED STRUCTURE

Native Link usage provides access to Compustat records, regardless of whether or not securities are in the CRSP universe. All Compustat data including index data, Canadian records, and off-exchange ranges that cannot be directly linked to CRSP Data are accessed using GVKEY, GVKEY.IID, and GVKEYX. The native link reads Compustat data as organized and identified by Compustat identifiers and can choose CRSP data appropriate to those records. Decisions on handling overlaps or soft links are left to the user.

Mnemonic	Field Name	Format
LINKDT	linkdt is a calendar date in YYYYMMDD format marking the first effective date of the current link. It is derived from the first or last date of a CRSP exchange listing, the date of a CRSP name change corresponding to the beginning or end of the link the rows of available Compustat data, or the date of a Compustat description change corresponding to the beginning or end of the link. If a linkdt is derived from a last date, it will actually be the day after the last date. Since CRSP keeps link records for the entire Compustat history, if the Compustat history ends after the CRSP history, the linkdt of a row marking a no-link period can start the day after the CRSP delist date.	integer
LINKENDDT	Last effective date of the link record. If the name represents current link information, the LINKENDDT is set to 99999999	integer
LPERMNO	CRSP PERMNO link during link period. It is set to zero if there is no CRSP link during the range.	integer
LPERMCO	CRSP PERMCO link during link period. It is set to zero if there is no CRSP link during the range.	integer
LIID	Security identifier	character
LNKTYPE	Link type code. Each link is given a code describing the connection between the CRSP and Compustat data. Values are: LC – Link research complete. Standard connection between databases. LU – Unresearched link to issue by CUSIP LX – Link to a security that trades on another exchange system not included in CRSP data. LD – Duplicate link to a security. Another GVKEY/IID is a better link to that CRSP record. LS – Link valid for this security only. Other CRSP PERMNOs with the same PERMCO will link to other GVKEYs. LN – Primary link exists but Compustat does not have prices. NR – No link available, confirmed by research NU – No link available, not yet confirmed	character
LINKPRIM	Primary issue marker for the link. Based on Compustat Primary/Joiner flag (PRIMISS), indicating whether this link is to Compustat's marked primary security during this range. P = Primary, identified by Compustat in monthly security data. J = Joiner secondary issue of a company, identified by Compustat in monthly security data. C = Primary, assigned by CRSP to resolve ranges of overlapping or missing primary markers from Compustat in order to produce one primary security throughout the company history. N = Secondary, assigned by CRSP to override Compustat. Compustat allows a US and Canadian security to both be marked as Primary at the same time. For Purposes of the link, CRSP allows only one primary at a time and marks the others as N.	character

LINKUSED DEFINED STRUCTURE

LINKUSED includes all of the fields in the link sturcture plus UGVKEY and USEDFLAG. Its number of rows exceeds that of the link structure because it has all link records of all GVKEYs with a match to a PERMNO, even those not used. USEDFLAG = 1 in LINKUSED is a subset of the available link records. It is loaded each time data are accessed in CRSP-Centric mode, such as for ts_print, TsQuery, or when C and FORTRAN functions are run. The function builds a composite Compustat record from one or more Compustat GVKEYs and IIDs linked to a CRSP PERMNO. LINKUSED data are accessed using the composite PERMNO, APERMNO, or the Primary PERMNO, PPERMNO.

Mnemonic	Field Name	Internal Storage
ULINKDT	ulinkdt is a calendar date in YYYYMMDD format marking the first effective date of the current link. It is derived from the first or last date of a CRSP exchange listing, the date of a CRSP name change corresponding to the beginning or end of the link the rows of available Compustat data, or the date of a Compustat description change corresponding to the beginning or end of the link.	integer
ULINKENDDT	Last effective date of the link record. If the name represents current link information, the ULINKENDDT is set to 99999999	integer
ULINKID	Unique ID per link associated with PERMNO. This is used to join with range data in the LINKRANGE table that describes the data ranges applied from used GVKEYs.	integer
UGVKEY	GVKEY used in the link	integer
UPERMNO	CRSP PERMNO link during link period. It is set to zero if there is no CRSP link during the range.	integer
UPERMCO	CRSP PERMCO link during link period. It is set to zero if there is no CRSP link during the range.	integer
UIID	Used Security ID	character
USEDFLAG	Flag marking whether link is used in building composite record	character
ULINKPRIM	Primary issue marker for the link. Based on Compustat Primary/Joiner flag (PRIMISS), indicating whether this link is to Compustat's marked primary security during this range. P = Primary, identified by Compustat in monthly security data. J = Joiner secondary issue of a company, identified by Compustat in monthly security data. C = Primary, assigned by CRSP to resolve ranges of overlapping or missing primary markers from Compustat in order to produce one primary security throughout the company history.	character
ULINKTYPE	Link type code. Each link is given a code describing the connection between the CRSP and Compustat data. Values are: LC – Link research complete. Standard connection between databases. LU – Unresearched link to issue by CUSIP. LX – Link to a security that trades on another exchange system not included in CRSP data. LD – Duplicate Link to a security. Another GVKEY/IID is a better link to that CRSP record. LS – Link valid for this security only. Other CRSP PERMNOs with the same PERMCO will link to other GVKEYs. LN – Primary link exists but Compustat does not have prices NR – No link available, confirmed by research NU – No link available, not yet confirmed	character

LINKRNG DEFINED STRUCTURE

CRSP generates a range table with information on the fiscal periods associated with each used link for each time series calendar frequency and keyset. This shows ranges in each of the fiscal and calendar calendars available in the CCM. This range table shows the ranges from the GVKEY for each type of time series data used to build the composite record for the PERMNO selected.

Mnemonic	Field Name	Format
RLINKID	Linkused row identifier	integer
RKEYSET	Keyset applicable to range	integer
RCALID	Calendar applicable to range	integer
RBEGIND	Beginning time series range of link	integer
RENDIND	Ending time series range of link	integer
RPREVIND	Time series range immediately prededing the link	integer
RBEGDT	Beginning calendar range of link	integer
RENDDT	Ending calendar range of link	integer
RPREVDT	Ending calendar range preceding the link	int(4)
RFISCAL_DATA_FLG	Type of time series, C-calendar or F-fiscal.	char(8)

CRSP-CENTRIC LINK USAGE

Accessing Compustat data through ts-print and TsQuery is done through the CRSP-centric mode, meaning that the primary access key in this mode is CRSP PERMNO or PERMCO rather than GVKEY, as used in the Native Access mode. The CRSP identifiers are the access keys while the Compustat identifiers become attributes.

In CRSP-Centric mode a composite record is built using the CRSP Link reading one or more GVKEYs. All GVKEYS with some presence of the PERMNO in the link are accessed. A used-link history is built from these link records by identifying those that cover the ranges of Compustat data needed to link to the CRSP identifier. The USEDFLAG for time series items that are stored on a fiscal period basis, the link ranges are translated to a fiscal range. This translation simplifies the selection of fundamental data that are applicable to the range and allows for the creation of a composite Compustat record from the applicable ranges that correspond to a CRSP security.

Records in CRSP-Centric form are identical in layout to the native records, but use CRSP PERMNO as the effective key. The Compustat component identifiers – GVKEY, IID, and PRIMISS are available in a Link Used table in the CRSP records.

Using the CRSP-Centric view simplifies access when viewing Compustat data through CRSP. One drawback, however, is that only data considered a primary link to CRSP, applied using CRSP link rules, are available.

Accessing two separate GVKEYs from the Link table, see that both share a single PERMNO.

GVKEY = 011947

```
Link History
------------
LINKDT    LINKENDDT        LPERMNO LPERMCO LIID LINKTYPE LINKPRIM
19820701  19860304        0         0         00X    NR            C
19860305  19890228        10083     8026     01     LU            P
```

GVKEY = 015495

```
Link History
------------
  LINKDT LINKENDDT               LPERMNO LPERMCO LIID LINKTYPE LINKPRIM
19880101  19890227        0          0      00X  NU              C
19890228  19930909        10083      8026 01   LC              C
19930910  19990304        0          0      01   NR           C
```

Using CRSP-Centric access in the USEDLINK table, access the composite history using the Primary PERMNO (LINKPRIM=P)

PERMNO = 10083

```
Link Used
---------
LINKDT LINKENDDT  GVKEY IID LINKID PERMNO PERMCO USEDFLAG LINKPRIM LINKTYPE
19820701  19860304  11947 00X      5      0      0       -1 C        NR
19860305  19890228  11947 01       6  10083   8026        1 P        LU
19880101  19890227  15495 00X      0      0      0       -1 C        NU
19890228  19930909  15495 01       1  10083   8026        1 C        LC
19930910  19990304  15495 01       2      0      0       -1 C        NR
19990305  20051019  15495 01       3  86787  16430       -1 C        LC
20051020  99999999  15495 01       4      0      0       -1 C        NR
```

Please read Chapter 2 for more complete coverage on the CRSP Link.

COMPANY DATA

ADJFACT DEFINED STRUCTURE

The ADJFACT structure contains company adjustment factor history.

Mnemonic	Field Name	Internal Storage
EFFDATE	Effective date- company cumulative factor	integer
THRUDATE	Thu date – company cumulative factor	integer
ADJEX	Cumulative adjustment factor by Ex-date	floating point double precision
ADJPAY	Cumulative adjustment factor by Pay-date	floating point double precision

HGIC DEFINED STRUCTURE

The HGIC structure contains company level GICS history.

Mnemonic	Field Name	Format
INDFROM	Effective from (start) date	integer
INDTHRU	Effective through (last) date	integer
GGROUPH	Industry group name	character
GINDH	Group industry	character

Mnemonic	Field Name	Format
GSECTORH	Group industry sector	character
GSUBINDH	Group sub-industries	character

OFFTITL DEFINED STRUCTURE

The OFFTITL structure contains company officer title data.

Mnemonic	Field Name	Format
OFID	Officer ID	integer
OFCD	Officer title	character
OFNM	Officer Name(s)	character

CCM_FILEDATE DEFINED STRUCTURE

The CCM_FILEDATE structure contains company filing date data.

Mnemonic	Field Name	Format
FDATADATE	Company filing data date	integer
FCONSOL	Company consolidation level filedate	character
FPOPSRC	Population source filedate	character
SRCTYPE	Document source type filedate	character
FILEDATE	Company filing date	integer

CCM_IPCD DEFINED STRUCTURE

The CCM_IPCD structure contains company industry presentation code data.

Mnemonic	Field Name	Format
IPDATADATE	Industry presentation code data date	integer
IPCONSOL	Level of consolidation (Industry presentation code)	character
IPPOPSRC	Population source (Industry presentation code)	character
IPCD	Industry presentation code	character

SECURITY DATA

SECURITY DEFINED STRUCTURE

The SECURITY structure contains security level header data.

Mnemonic	Field Name	Format
EXCHG	Stock exchange	integer
DLDTEI	Security inactivation date	integer
IID_SEQ_NUM	IID sequence number	integer
SBEGDT	First date of Compustat data for issue	integer
SENDDT	Last date of Compustat data for issue	integer
IID	Issue ID	character
SCUSIP	CUSIP	character
DLRSNI	Security inactivation code	character
DSCI	Security description	character
EPF	Earnings participation flag	character

Mnemonic	Field Name	Format
EXCNTRY	Stock exchange country code	character
ISIN	International security identification number	character
SSECSTAT	Security status marker	character
SEDOL	SEDOL	character
TIC	Ticker/trading symbol	character
TPCI	Issue type	character

SECHIST DEFINED STRUCTURE

The SECHIST structure contains security header history data.

Mnemonic	Field Name	Format
HSCHGDT	Historical security change date	integer
HSCHGENDDT	Historical security change end date	integer
HEXCHG	Historical stock exchange	integer
HDLDTEI	Historical security inactivation date	integer
HIID_SEQ_NUM	Historical issue ID sequence number	integer
HIID	Historical issue ID	character
HSCUSIP	Historical CUSIP	character
HDLRSNI	Historical security inactivation code	character
HDSCI	Historical security description	character
HEPF	Historical earnings participation flag	character
HEXCNTRY	Historical stock exchange country code	character
HISIN	Historical international security identification number	character
HSSECSTAT	Historical security status marker	character
HSEDOL	Historical SEDOL	character
HTIC	Historical ticker/trading symbol	character
HTPCI	Historical issue type	character

SEC_MTHSPT DEFINED STRUCTURE

The SEC_MTHSPT structure contains security monthly split event data.

Mnemonic	Field Name	Format
DATADATEM	Monthly adjustment factor data date	integer
RAWPM	Raw adjustment factor – pay date - monthly	floating point double precision
RAWXM	Raw adjustment factor – ex date - monthly	floating point double precision

SEC_MSPT_FN DEFINED STRUCTURE

The SEC_MSPT_FN structure contains security monthly split event footnotes.

Mnemonic	Field Name	Format
DATADATEMF	Monthly adjustment factor footnote data date	integer
DATAITEMMF	Monthly split footnote dataitem	character
RAWPM_FN1..FN5	Raw adjustment factor – pay date – monthly – footnotes 1-5	character
RAWXM_FN1..FN5	Raw adjustment factor – ex date – monthly – footnotes 1-5	character

SEC_MDIV_FN DEFINED STRUCTURE

The SEC_MDIV_FN structure contains security monthly dividend event footnotes.

Mnemonic	Field Name	Format
DIVDATADATEMF	Monthly dividend footnote data date	integer
DIVDATAITEMMF	Monthly dividend footnote data item	character
DVPSPM_FN1..FN5	Dividend per share – pay date – monthly – footnotes 1-5	character
DVPSXM_FN1..FN5	Dividend per share – ex date – monthly – footnotes 1-5	character

SEC_SPIND DEFINED STRUCTURE

The SEC_SPIND structure contains data associated with security S&P Industry events.

Mnemonic	Field Name	Format
SPBEGDATE	S&P Index event beginning date	integer
SPENDDATE	S&P Index event ending date	integer
SPHIID	S&P holdings industry index ID	integer
SPHMID	S&P holdings major index ID	integer
SPHSEC	S&P holdings sector code	integer
SPH100	S&P holdings S&P 100 marker	integer
SPHCUSIP	S&P holdings CUSIP	character
SPHNAME	S&P holdings name	character
SPHTIC	S&P holdings ticker	character
SPHVG	S&P holdings value/growth indicator	character

IDXCST_HIS DEFINED STRUCTURE

The IDXCST_HIS structure contains security historical industry constituent data.

Mnemonic	Field Name	Format
XFROM	S&P constituent from event date	integer
IDX13KEY	S&P 13 character key	character
XGVKETX	S&P constituent event index GVKEY	integer

SEGMENT DATA

CCM_SEGCUR DEFINED STRUCTURE

The CCM_SEGCUR structure contains operating segment currency rate data.

Mnemonic	Field Name	Format
SC_DATAYR	Data year	integer
SC_DATAFYR	Data fiscal year end month	integer
SC_CALYR	Data calendar year	integer
SC_SRCYFYR	Source fiscal year end month	integer
SC_XRATE	Period end exchange rate	floating point double precision
SC_XRATE12	12-month moving exchange rate	floating point double precision
SC_SRCCUR	Source currency code	character
SC_CURCD	ISO currency code (USD)	character

CCM_SEGSRC DEFINED STRUCTURE

The CCM_SEGSRC structure contains operating segment source data.

Mnemonic	Field Name	Format
SS_SRCYR	Source year	integer
SS_SRCFYR	Source fiscal year end month	integer
SS_CALYR	Data calendar year	integer
SS_RCST1	Reserved 1	integer
SS_SSRCE	Source document code	character
SS_SUCODE	Source update code	character
SS_CURCD	ISO currency code	character
SS_SRCCUR	Source ISO currency code	character
SS_HNAICS	Segment primary historical NAICS	character

CCM_SEGPROD DEFINED STRUCTURE

The CCM_SEGPROD structure contains operating segment product data.

Mnemonic	Field Name	Format
SP_SRCYR	Source year	integer
SP_SRCFYR	Source fiscal year end month	integer
SP_PDID	Product identifier	integer
SP_PSID	Segment link – segment identifier	integer
SP_PSALE	External revenues	floating point single precision
SP_RCST1	Reserved 1	floating point single precision
SP_PNAICS	Product NAICS code	character
SP_PSTYPE	Segment link- segment type	character
SP_PNAME	Product name	character

CCM_SEGCUST DEFINED STRUCTURE

The CCM_SEGCUST structure contains operating segment customer data.

Mnemonic	Field Name	Format
SC_SRCYR	Source year	integer
SC_SRCFYR	Source fiscal year end month	integer
SC_CDID	customer identifier	integer
SC_CSID	Segment link – segment identifier	integer
SC_CSALE	Customer revenues	floating point single precision
SC_RCST1	Reserved 1	integer
SC_CTYPE	Customer type	character
SC_CGEOCD	Geographic area code	character
SC_CGEOAR	Geographic area type	character
SC_CSTYPE	Segment link – segment type	character
SC_CNAME	Customer name data	character

CCM_SEGDTL DEFINED STRUCTURE

The CCM_SEGDTL structure contains operating segment detail data.

Mnemonic	Field Name	Format
SD_SRCYR	Source year	integer
SD_SRCFYR	Source fiscal year end month	integer
SD_SID	Segment identifier	integer
SD_RCST1	Reserved 1	integer
SD_STYPE	Segment type	character
SD_SOPTP1	Operating segment type 1	character
SD_SOPTP2	Operating segment type	character
SD_SGEOTP	Geographic segment type	character
SD_SNAME	Segment name	character

CCM_SEGITM DEFINED STRUCTURE

The CCM_SEGITM structure contains operating segment item data.

Mnemonic	Field Name	Format
SI_DATYR	Data year	integer
SI_FISCYR	Data fiscal year end month	integer
SI_SRCYR	Source year	integer
SI_SRCFYR	Source fiscal year end month	integer
SI_CALYR	Data calendar year	integer
SI_SID	Segment identifier	integer
SI_EMP	Employees	integer
SI_SALE	Net sales	floating point single precision
SI_OIBD	Operating income before depreciations	floating point single precision
SI_DP	Depreciation & amortization	floating point single precision
SI_OIAD	Operating income after depreciation	floating point single precision
SI_CAPX	Capital expenditures	floating point single precision
SI_IAT	Identifiable assets	floating point single precision
SI_EQEARN	Equity in earnings	floating point single precision
SI_INVEQ	Investments at equity	floating point single precision
SI_RD	Research & development	floating point single precision
SI_OBKLG	Order backlog	floating point single precision
SI_EXPORTS	Export sales	floating point single precision
SI_INTSEG	Inter-segment eliminations	integer
SI_OPINC	Operating profit	floating point single precision
SI_PI	Pretax income	floating point single precision
SI_IB	Income before extraordinary earnings	floating point single precision
SI_NI	Net income (loss)	floating point single precision
SI_RCST1	Reserved 1	floating point single precision
SI_RCST2	Reserved 2	floating point single precision
SI_RCST3	Reserved 3	floating point single precision
SI_SALEF	Footnote 1 - sales	character
SI_OPINCF	Footnote 2 – operating profit	character

Mnemonic	Field Name	Format
SI_CAPXF	Footnote 3 – capital expenditures	character
SI_EQEARNF	Footnote 4 – equity in earnings	character
SI_EMPF	Footnote 5 - employees	character
SI_RDF	Footnote 6 – research & development	character
SI_STYPE	Segment type	character

CCM_SEGNAICS DEFINED STRUCTURE

The CCM_SEGNAICS structure contains operating segment NAICS data.

Mnemonic	Field Name	Format
SN_SRCYR	Source year	integer
SN_SRCFYR	Source fiscal year end month	integer
SN_SID	Segment identifier	integer
SN_RCST1	Reserved 1	integer
SN_STYPE	Segment type	character
SN_SNAICS	NAICS code	character
SN_RANK	Ranking	integer
SN_SIC	Segment SIC code	integer

CCM_SEGGEO DEFINED STRUCTURE

The CCM_SEGGEO structure contains operating segment geographic data.

Mnemonic	Field Name	Format
SG_SRCYR	Source year	integer
SG_SRCFYR	Source fiscal year end month	integer
SG_SID	Segment identifier	integer
SG_RCST1	Reserved 1	integer
SG_STYPE	Segment type	character
SG_SGEOCD	Geographic area code	character
SG_SGEOTP	Geographic area type	character

KEYSETS

Compustat items can be qualified by a set of secondary keys. This collection of secondary keys and values create a keyset that assigns a numeric code and mnemonic tag to each unique collection. Each keyset represents different output series. When multiple keysets are available for a particular data item, users can specify both the item and keyset to identify the series of interest or simply use the default preset combination that is most commonly used.

For example, the data item SALE has secondary keys for industry format, data format, population source, and consolidation level. A different value of company sales may be available for any combination of these keys. One keyset may represent originally reported sales. Another may represent the final restated sales from a later filing.

Keyset	Tag	Keyset Components	Keyset Description
All Keysets use a Domestic POPSRC and use some form of standardized data in their DATAFMT presentation			
0		Null Keyset, no variations using secondary keys	Null Keyset, no variations using secondary keys
1	STD	DATAFMT = STD INDFMT = INDL CONSOL = C POPSRC = D	Industrial Format, Consolidated Information, Standardized Presentation
2	SUMM	DATAFMT = SUMM_STD INDFMT = INDL CONSOL = C POPSRC = D	Industrial Format, Consolidated Information, Standardized Summary Data from the Latest Annual Filing
3	PRES	DATAFMT = PRE_AMENDSS INDFMT = INDL CONSOL = C POPSRC = D	Industrial Format, Consolidated Information, Standardized Summary Data Collected prior to Company Amendment
4	FS	DATAFMT = STD INDFMT = FS CONSOL = C POPSRC = D	Financial Services Format, Consolidated Information, Standardized Presentation
5	PFO	DATAFMT = STD INDFMT = INDL CONSOL = R POPSRC = D	Industrial Format, Pro Forma Reporting, Standardized Presentation
6	PFAS	CONSOL = P POPSRC = D	Pre FASB Reporting
7	SFAS	DATAFMT = STD INDFMT = INDL CONSOL = P POPSRC = D	Industrial Format, Pre-FASB Reporting, Standardized Presentation
8	PRE	DATAFMT = PRE_AMENDS INDFMT = INDL CONSOL = C POPSRC = D	Industrial Format, Consolidated Information, Standardized Data Collected from the Latest Annual Filing
10	PDIV	DATAFMT = STD INDFMT = INDL CONSOL = D POPSRC = D	Industrial Format, Pre-Divestiture Reporting, Standardized Presentation
11	DOM	POPSRC = D	Domestic
12	SUPF	DATAFMT = SUMM_STD INDFMT = INDL CONSOL = P POPSRC = D	Industrial Format, Pre-FASB Reporting, Standardized Summary Data from the Latest Annual Filing
14	STD1	DATAFMT = STD INDFMT = INDL CONSOL = C POPSRC = D RANK = 1	Industrial Format, Consolidated Information, Standardized Presentation, Rank 1
15	FSFO	DATAFMT = STD INDFMT = FS CONSOL = R POPSRC = D	Financial Services Format, Pro-Forma Reporting, Standardized Presentation
16	FS1	DATAFMT = STD INDFMT = FS CONSOL = C POPSRC = D RANK = 1	Financial Services Format, Consolidated Information, Standardized Presentation, Rank 1
17	FS2	DATAFMT = STD INDFMT = FS CONSOL = C POPSRC = D RANK = 2	Financial Services Format, Consolidated Information, Standardized Presentation, Rank 2
18	SUFS	DATAFMT = SUMM_STD INDFMT = INDL CONSOL = R POPSRC = D	Industrial Format, Pro-Forma Reporting, Standardized Summary Data from the Latest Annual Filing
19	PDI1	DATAFMT = STD INDFMT = INDL CONSOL = D POPSRC = D RANK = 1	Industrial Format, Pre-Divestiture Reporting, Standardized Presentation, Rank 1
20	PFA1	DATAFMT = STD INDFMT = INDL CONSOL = P POPSRC = D RANK = 1	Industrial Format, Pre-FASB Reporting, Standardized Presentation, Rank 1
21	SUPD	DATAFMT = SUMM_STD INDFMT = INDL CONSOL = D POPSRC = D	Industrial Format, Pre-Divestiture Reporting, Standardized Summary Data from the Latest Annual Filing
22	FS3	DATAFMT = STD INDFMT = FS CONSOL = C POPSRC = D RANK = 3	Financial Services Format, Consolidated Information, Standardized Presentation, Rank 3
23	PDI2	DATAFMT = STD INDFMT = INDL CONSOL = D POPSRC = D RANK = 2	Industrial Format, Consolidated Information, Standardized Presentation, Rank 2
24	CONS	CONSOL = C POPSRC = D	Consolidated Information
25	STD2	DATAFMT = STD INDFMT = INDL CONSOL = C POPSRC = D RANK = 2	Industrial Format, Consolidated Information, Standardized Presentation, Rank 2

Keyset	Tag	Keyset Components	Keyset Description
26	STD3	DATAFMT = STD INDFMT = INDL CONSOL = C POPSRC = D RANK = 3	Industrial Format, Consolidated Information, Standardized Presentation, Rank 3
27	STD4	DATAFMT = STD INDFMT = INDL CONSOL = C POPSRC = D RANK = 4	Industrial Format, Consolidated Information, Standardized Presentation, Rank 4
28	STD5	DATAFMT = STD INDFMT = INDL CONSOL = C POPSRC = D RANK = 5	Industrial Format, Consolidated Information, Standardized Presentation, Rank 5
29	PFA2	DATAFMT = STD INDFMT = INDL CONSOL = P POPSRC = D RANK = 2	Industrial Format, Pre-FASB Reporting, Standardized Presentation, Rank 2
30	PFA3	DATAFMT = STD INDFMT = INDL CONSOL = P POPSRC = D RANK = 3	Industrial Format, Pre-FASB Reporting, Standardized Presentation, Rank 3
31	CUSD	CFFLAG = C POPSRC = D MKT_CURCD = USD	Calendar Based Reporting in US Dollars
32	FUSD	CFFLAG = F POPSRC = D MKT_CURCD = USD	Fiscal Based Reporting in US Dollars
33	CCAD	CFFLAG = C POPSRC = D MKT_CURCD = CAD	Calendar Based Reporting in Canadian Dollars
34	FCAD	CFFLAG = F POPSRC = D MKT_CURCD = CAD	Fiscal Based Reporting in Canadian Dollars
35	PFA4	DATAFMT = STD INDFMT = INDL CONSOL = P POPSRC = D RANK = 4	Industrial Format, Pre-FASB Reporting, Standardized Presentation, Rank 4
36	PFO2	DATAFMT = STD INDFMT = INDL CONSOL = R POPSRC = D RANK = 2	Industrial Format, Pro-Forma Reporting, Standardized Presentation, Rank 2
37	PFO1	DATAFMT = STD INDFMT = INDL CONSOL = R POPSRC = D RANK = 1	Industrial Format, Pro-Forma Reporting, Standardized Presentation, Rank 1
38	PRE1	DATAFMT = PRE_AMENDS INDFMT = INDL CONSOL = C POPSRC = D RANK = 1	Industrial Format, Consolidated Information, Standardized Data Collected before Company Amendment, Rank 1
39	FFO1	DATAFMT = STD INDFMT = FS CONSOL = R POPSRC = D RANK = 1	Financial Services Format, Pro-Forma Reporting, Standardized Presentation, Rank 1
40	FS4	DATAFMT = STD INDFMT = FS CONSOL = C RANK = 4	Financial Services format, Consolidated Information, Standardized Presentation, Rank 4
41	GICS	INDTYPE = GICS	Industry Code Type GICS
43	FORD	CONSOL = R POPSRC = D	Pro-Forma Reporting
45	POSTDIV	DATAFMT = SUMM_STD INDFMT = BANK CONSOL = C POPSRC = D	Industrial Format, Post-Divestiture Reporting, Standardized Presentation
46	POSTDIV1	DATAFMT = STD INDFMT = BANK CONSOL = R POPSRC = D	Industrial Format, Post-Divestiture Reporting, Standardized Presentation, Rank 1
2100	BSTD	DATAFMT = STD INDFMT = INDL POPSRC = D CONSOL = C	Bank Format, Consolidated Information, Standardized Presentation
2101	BSUMM	DATAFMT = STD INDFMT = INDL POPSRC = D CONSOL = C	Bank Format, Consolidated Information, Standardized Summary Data from the Latest Annual Filing
2102	BSTD1	CONSOL=C DATAFMT=STD INDFMT=BANK POPSRC=D RANK = 1	Bank Format, Consolidated Information, Standardized Presentation, Rank 1
2103	BSTD2	DATAFMT = STD INDFMT = INDL POPSRC = D CONSOL = C	Bank Format, Consolidated Information, Standardized Presentation, Rank 2
2120	BASTD	DATAFMT = STD INDFMT = INDL POPSRC = D CONSOL = C	Bank Format, Consolidated Information, Average Standardized Presentation
2121	BASUMM	DATAFMT = STD INDFMT = INDL POPSRC = D CONSOL = C	Bank Format, Consolidated Information, Average Standardized Summary Presentation from the Latest Annual Filing
2122	BASTD1	DATAFMT = STD INDFMT = INDL POPSRC = D CONSOL = C	Bank Format, Consolidated Information, Average Standardized Presentation, Rank 1

Keyset	Tag	Keyset Components	Keyset Description
2123	BASTD2	DATAFMT = STD INDFMT = INDL POPSRC = D CONSOL = C	Bank Format, Consolidated Information, Average Standardized Presentation, Rank 2
2140	BPFO	DATAFMT = STD INDFMT = INDL POPSRC = D CONSOL = C	Bank Format, Pro-Forma Reporting, Standard Presentation
2160	BAPFO	DATAFMT = STD INDFMT = INDL POPSRC = D CONSOL = C	Bank Format, Pro-Forma Reporting, Average Standardized Presentation

MISSING DATA

NOTES ON MISSING VALUES

Compustat provides specific codes for data items which are:

- Not available
- Combined with other data items
- Considered insignificant by the company
- Available only on a semi-annual or annual basis

The data provided in the CRSPAccess format have constants representing each case. Missing value codes conform with Compustat's Strategic Insight and binary conventions for missing values.

Compustat Missing Value Codes		
Missing Value Code	**C Constant**	**Numeric Value**
No data for data item	CST_MISS_NA	.0001
Data has been combined into another item	CST_MISS_COMB	.0004
Data has been reported by the company as insignificant	CST_MISS_INSIG	.0008
Data is only reported on a semi-annual basis	CST_MISS_SEMI	.0002
Data is only reported on an annual basis	CST_MISS_ANN	.0003

CHAPTER 2: CRSP LINK®

OVERVIEW

CRSP and Compustat data are commonly linked to match CRSP event and market data history with Compustat fundamental and supplemental data. Because of different identification conventions, universes, available historical information, and conventions unique to each organization, linking is not a straightforward process. Through using the CRSP Link®, a data array which contains a history of links using CRSP and Compustat identifiers, subscribers may accurately combine CRSP and Compustat data into a single source of clean, reliable data.

Compustat Xpressfeed provides new security level data requiring adjustments to the linking process between CRSP and Compustat databases. Previously, Compustat included one security per record. Now all securities are available with a new identifier, IID, which can be used along with GVKEY to permanently identify all securities tracked by Compustat, and marker items that identify the security that Compustat considers Primary.

CRSP provides two views of the data through the CRSP Link. While the standard form is the native data and linking information that is organized by Compustat GVKEY, CRSP provides tools to use the link to build CRSP-centric records linked by PERMNO.

Identifiers used by the link:

GVKEY Compustat's permanent company identifier.

IID Compustat's permanent issue identifier. An identifying relationship exists between IID and GVKEY. Both must be accessed as a pair to properly identify a Compustat security. One GVKEY can have multiple IIDs.

Because Compustat company data ranges can extend earlier than security ranges, there may be some time periods with no identified IID for a GVKEY. In these cases, CRSP assigns a dummy IID ending in "X" as a placeholder in the link. This range may or may not be associated with a CRSP PERMNO, but there is no Compustat security data found during the range when no IID is assigned.

PRIMISS Compustat provides a primary marker indicating which security is considered primary for a company at a given time.

PERMCO CRSP's permanent company identifier.

PERMNO CRSP's permanent issue identifier. One PERMNO belongs to only one PERMCO. One PERMCO can have one or more PERMNOs.

THE LINKING PROCESS

Prior to the introduction of Xpressfeed, Compustat included only one security per record. The links between CRSP and Compustat were between CRSP PERMNO and Compustat GVKEY. Because PERMNO is a security identifier and GVKEY is a company identifier, the linking could be a many to one relationship. More than one PERMNO may be linked to a single GVKEY.

CRSP addressed the security links in phases. The initial phase addressed security links for issues after mid-April in 2007, for that was when the first Compustat security-level information was available. In this phase, links prior to this time were maintained by using the old CST link information as a foundation onto which updates and refinements were applied.

The primary goal of the second phase of building the security links was to remove the April 2007 starting limitation to the security-based links and move to a full security link history. Once the full security history was built, it would be used to generate company –based historical linking broken down into primary issue ranges and indicators.

This process is laborious and demanding of CRSP researchers and programmers. The new links are reflected beginning with the release of the 200806 annual (CMX200806) and the 200810 monthly and quarterly (CMX200810) release.

NATIVE LINK ACCESS

The native link, accessing data using GVKEY, GVKEY.IID, and GVKEYX is used to access all Compustat data including index data, Canadian records, and off-exchange ranges that cannot be directly linked to the CRSP securities. The native link reads Compustat data as organized and identified by Compustat identifiers and can choose CRSP data associated with those records. Decisions on handling overlaps or soft links are left to the user.

CRSP provides security level link data with a flag, PRIMFLAG, indicating whether or not each link is to Compustat's identified primary issue. The primary issue flag can be used to restrict the link to one security per company for each range as it was done with the original CRSP link. Primary issue flags are P, Primary as identified by Compustat, or C, Primary assigned by CRSP.

Example: Accessing two separate GVKEYs in Native Mode from the Link table, see that both share a single PERMNO.

GVKEY = 011947

```
Link History
------------
LINKDT   LINKENDDT LPERMNO LPERMCO  LIID LINKTYPE LINKPRIM
19820701 19860304        0       0   00X    NR       C
19860305 19890228    10083    8026    01    LU       P
```

GVKEY = 015495

```
Link History
------------
LINKDT   LINKENDDT LPERMNO LPERMCO  LIID LINKTYPE LINKPRIM
19880101 19890227        0       0   00X    NR       C
19890228 19930909    10083    8026    01    LC       C
19930910 19990304        0       0    01    NR       C - Delisted
```

CRSP_CCM_LINK – SECURITY LINK HISTORY

Only one set of link information is presented for each calendar range in the Compustat GVKEY and IID history. Soft LX and LD links are included if there is a match that indicates an alternate record or a security on a non-US exchange. CRSP provides no automated methods to use these soft links to connect to CRSP data, but the information is available for the user.

Native Link usage provides access to all Compustat records, regardless of whether or not securities are in the CRSP universe.

Itm_name	Type	Description
GVKEY*	integer, primary key (1)	Compustat GVKEY
LIID	char(3), primary key (2)	Compustat IID. Dummy IID assigned with an "X" suffix during a range when company data exists but no Compustat security is identified.
LINKDT	integer (date), primary key (3)	First effective calendar date of link record range
LINKENDDT	integer (date)	Last effective calendar date of link record range
LPERMNO	integer	Linked CRSP PERMNO, 0 if no CRSP security link exists
LPERMCO	integer	Linked CRSP PERMCO, 0 if no CRSP company link exists

Itm_name	Type	Description
LINKPRIM	char(3)	Primary issue marker for the link. Based on Compustat Primary/Joiner flag (PRIMISS), indicating whether this link is to Compustat's marked primary security during this range. P = Primary, identified by Compustat in monthly security data. J = Joiner secondary issue of a company, identified by Compustat in monthly security data. C = Primary, assigned by CRSP to resolve ranges of overlapping or missing primary markers from Compustat in order to produce one primary security throughout the company history. N = Secondary, assigned by CRSP to override Compustat. Compustat allows a US and Canadian security to both be marked as Primary at the same time. For Purposes of the link, CRSP allows only one primary at a time and marks the others as N.
LINKTYPE	char(3)	Link type code. Each link is given a code describing the connection between the CRSP and Compustat data. Values are: LC – Link research complete. Standard connection between databases. LU – Unresearched link to issue by CUSIP. (LU ignores LD.) LX – Link to a security that trades on foreign exchange not included in CRSP data. LD – Duplicate Link to a security. Another GVKEY/IID is a better link to that CRSP record. (2 GVKEYs - duplicate record) LN – Primary link exists but Compustat does not have prices. LS – Link valid for this security only. Other CRSP PERMNOs with the same PERMCO will link to other GVKEYs. (Happens frequently for ETFs.) NR – No link available, confirmed by research NU – No link available, not yet confirmed

** - The GVKEY is the primary key of all Compustat company records when using the native link. In CRSPAccess programming this field is not present in the structure but inherited from the CCMID item in the master structure for the company. In standalone usage the GVKEY field is included.*

CRSP-CENTRIC LINK USAGE

Accessing Compustat data through ts_print and TsQuery is done through the CRSP-centric mode, meaning that the primary access key in this mode is CRSP PERMNO rather than GVKEY, as used in the Native Access mode. The CRSP identifiers are the access keys while the Compustat identifiers become attributes. There are two options: Primary only, which mirrors the company-level link by ignoring links not to the primary security, and All, which allows use of any link to the PERMNO.

In CRSP-Centric mode a composite record is built using the CRSP Link reading one or more GVKEYs. All GVKEYS with some presence of the PERMNO in the link are accessed. A used-link history is built from these link records by identifying those that cover the ranges of Compustat data needed to link to the CRSP identifier. For time series items that are stored on a fiscal period basis, the link ranges are translated to a fiscal range. This translation simplifies the selection of fundamental data that are applicable to the range and allows for the creation of a composite Compustat record from the applicable ranges that correspond to a CRSP security.

Records in CRSP-Centric form are identical in layout to the native records, but use CRSP PERMNO as the effective key. The Compustat component identifiers – GVKEY, IID, and PRIMISS are available in a Link Used table in the CRSP records.

Using the CRSP-Centric view simplifies access when viewing Compustat data through CRSP. One drawback, however, is that only data considered a direct link to CRSP, applied using CRSP link rules, are available.

The example that follows accessed data natively, then through the CRSP-centric view using PERMNO.

Example: Accessing two separate GVKEYs from the Link table, see that both share a single PERMNO.

GVKEY = 011947

```
Link History
------------
LINKDT   LINKENDDT     LPERMNO LPERMCO LIID LINKTYPE LINKPRIM
19820701 19860304      0       0       00X     NR       C
19860305 19890228      10083   8026     01     LU       P
```

GVKEY = 015495

```
Link History
------------
LINKDT   LINKENDDT     LPERMNO LPERMCO LIID LINKTYPE LINKPRIM
19880101 19890227      0       0       00X     NU       C
19890228 19930909      10083   8026    01      LC       C
19930910 19990304      0       0       01      NR       C
```

Using CRSP-Centric access, the LINKUSED data show which GVKEYs and IIDs are used to build a composite record by PERMNO. Only the rows with USEDFLAG=1 show the GVKEYs and calendar ranges used to build the composite record for PERMNO 10083. table, access the composite history using the Primary PERMNO (LINKPRIM=P)

PERMNO = 10083

```
Link Used
---------
LINKDT   LINKENDDT GVKEY IID LINKID PERMNO PERMCO USEDFLAG LINKPRIM LINKTYPE
19820701 19860304  11947 00X    5       0      0   -1        C        NR
19860305 19890228  11947 01     6   10083   8026    1        P        LU
19880101 19890227  15495 00X    0       0      0   -1        C        NU
19890228 19930909  15495 01     1   10083   8026    1        C        LC
19930910 19990304  15495 01     2       0      0   -1        C        NR
19990305 20051019  15495 01     3   86787  16430   -1        C        LC

20051020 99999999  15495 01     4       0      0   -1        C        NR
```

CRSP_CCM_LINKUSED – CRSP-CENTRIC LINK USED HISTORY

Itm_name	Type	Description
PERMNO*	integer, primary key (1)	CRSP PERMNO used as basis for this history
ULINKID	integer	Unique ID per link associated with PERMNO. This is used to join with range data in the LINKRANGE table that describes the data ranges applied from used GVKEYs.
UGVKEY	integer	Compustat GVKEY
UIID	char(3)	Compustat IID
ULINKDT	integer (date), primary key (2)	First effective calendar date of link record range
ULINKENDDT	integer (date)	Last effective calendar date of link record range
UPERMNO	integer	Linked CRSP PERMNO, 0 if no CRSP security link exists
UPERMCO	integer	Linked CRSP PERMCO, 0 if no CRSP company link exists

Itm_name	Type	Description
ULINKPRIM	char(3)	Primary issue marker for the link. Based on Compustat Primary/Joiner flag (PRIMISS), indicating whether this link is to Compustat's marked primary security during this range. P = Primary, identified by Compustat in monthly security data. J = Joiner secondary issue of a company, identified by Compustat in monthly security data. C = Primary, assigned by CRSP to resolve ranges of overlapping or missing primary markers from Compustat in order to produce one primary security throughout the company history. N = Secondary, assigned by CRSP to override Compustat. Compustat allows a US and Canadian security to both be marked as Primary at the same time. For Purposes of the link, CRSP allows only one primary at a time and marks the others as N.
ULINKTYPE	char(3)	Link type code. Each link is given a code describing the connection between the CRSP and Compustat data. Values are: LC – Link research complete. Standard connection between databases. LU – Unresearched link to issue by CUSIP. (LU ignores LD.) LX – Link to a security that trades on foreign exchange not included in CRSP data. LD – Duplicate Link to a security. Another GVKEY/IID is a better link to that CRSP record. (2 GVKEYs - duplicate record.) LN – Primary link exists but Compustat does not have prices. LS – Link valid for this security only. Other CRSP PERMNOs with the same PERMCO will link to other GVKEYs. (Happens frequently for ETFs.) NR – No link available, confirmed by research NU – No link available, not yet confirmed
USEDFLAG	integer	1 = this link is applicable to the selected PERMNO and used to identify ranges of Compustat data from a GVKEY used to build a composite GVKEY record corresponding to the PERMNO. -1 = this link is informational, indirectly related to the PERMNO, but not used.

** - The PERMNO is the CRSP security identifier used as the basis for a composite Compustat record and serves as the primary identifier for the composite record. In CRSPAccess programming this field is not present in the structure but inherited from the master structure for the company. The APERMNO or PPERMNO key types store the PERMNO in the CCM structure CCMID field and marks the CCMIDTYPE as 3. In standalone usage the PERMNO field is included.*

CRSP_CCM_LINKRNG – CRSP-CENTRIC LINK HISTORY RANGE

The link history is presented by calendar range. If data are presented on a fiscal basis the calendar dates must be interpreted as the proper fiscal period. In this case there can be overlaps generated when links change across GVKEYS or fiscal year end month changes.

CRSP generates a range table with information on the fiscal periods associated with each used link for each time series calendar frequency and keyset. This shows ranges in each of the fiscal and calendar calendars available in the CCM. When there is an overlap and used links provide data for the same fiscal period, the link with the latest filing data date is chosen for the fiscal period. This range table shows the ranges from the GVKEY for each type of time series data used to build the composite record for the PERMNO selected.

Itm_name	Type	Description
PERMNO*	integer, primary key 1	PERMNO key built
RLINKID	integer, primary key 2	unique ID set in the link used record, used for joining range data with the appropriate link
RKEYSET	integer, primary key 3	Keyset of time series object
RCALID	integer, primary key 4	CRSP calendar of time series

Itm_name	Type	Description
RFISCAL_ DATA_FLG	char(1)	Type of time series data, F = fiscal, C= calendar
RBEGIND	integer	first index in time series with valid data for this used link
RENDIND	integer	last index in time series with valid data for this used link
RPREVIND	integer	index of previous data
RBEGDT	integer	first calendar date in time series with valid data for this used link.
RENDDT	integer	last calendar date in time series with valid data for this used link
RPREVDT	integer	date of previous data

** - see note on CRSP_CCM_LINKUSED PERMNO.*

LINK ACTIONS

This table shows the types of links that are supported by the CRSP CCM link and how they are achieved. A date range is associated with each link so all actions imply an event history.

#	Action	Input Identifier Type	Output Identifier Type	Link Table
1	Find all securities in CRSP for Compustat Company data	GVKEY	PERMNO (PERMCO)	crsp_ccm_link (all links used)
2	Find primary security in CRSP for Compustat Company data	GVKEY	PERMNO	crsp_ccm_link (only links where LINKPRIM is P or C)
3	Find data in CRSP for a specific Compustat Company and issue	GVKEY/IID	PERMNO	crsp_ccm_link (links with desired IID)
4	Find Compustat data for a given CRSP security	PERMNO	GVKEY/IID	crsp_ccm_linkused (history used to build a composite GVKEY record in link used)
5	Find Compustat company and security data for a CRSP security, only if it is considered primary	PERMNO	GVKEY/IID	crsp_ccm_linkused (only use links where LINKPRIM is P or C)

LINK ACTION NOTES:

1. CRSP_CCM_LINK contains valid links for all securities provided by Compustat. Each record with a valid link to a PERMNO can be followed to the appropriate CRSP data. The user has the option of restricting links by LINKTYPE to ignore soft links, and using the CRSP PERMCO to identify other issues of the same company not addressed in the link. All PERMNOs found with this method share the company-level data from the GVKEY. The link record IID is needed to match the CRSP PERMNO data to the proper Compustat security level data.
2. Link records with the security not marked Primary are ignored. Otherwise this is the same as #1. The result is that even if multiple CRSP PERMNOs are found, there should be no overlap in the CRSP history used. All PERMNOs found will share the company-level data from the GVKEY, but will match only the Compustat IID indicated in the link record.
3. Given a GVKEY and IID from Compustat, use CRSP_CCM_LINK to get the history of CRSP PERMNOs linked to that company and security. The user has the option of restricting soft links using LINKTYPE. No consideration is given to whether the security is considered primary any time during its history. The link can produce multiple CRSP PERMNOs, but only one link should be found at any time.
4. Given a CRSP PERMNO, use CRSP_CCM_LINKUSED to find Compustat data. Access with APERMNO key type will build a composite GVKEY record from the used link records. CRSP_CCM_LINKRNG is used to find ranges of data for the composite record. Secondary links are ignored, and only the Compustat security data matching the permno are included.

There will be one composite security record created with a pseudo IID of 01X.

5. Same as #4, but a link record is ignored if the security matched is not primary. This will result in a smaller range, and a not-found if the PERMNO is never primary for the company. Access with PPERMNO key type is used to select this method.

6. PERMCO is not directly supported with linkused, but attached PERMNOs can be found from the PERMCO and the user can select securities with PERMNO. To avoid double-counting company data, the primary flag can be used to ensure that only one security is represented during each time range.

4,5. A user can use secondary index on PERMNO or PERMCO to find GVKEYs with matching information and see the Compustat data in native form, then handle processing as desired. These reads are not necessarily unique, so it is left to the user to select information from the correct ranges corresponding to the desired CRSP identifier.

TABLE VS. CRSPACCESS USAGE NOTES

The Link Actions table includes the primary identifiers for the databases: GVKEY for CCM and PERMNO for CRSP Stock. In a standalone setup where data are dumped and stored as a table these identifiers are included in each table and used to join data.

CRSPAccess programming access always organizes all data for one GVKEY (CCM) or PERMNO (CRSP Stock) in a single structure. The primary identifier is set at the full structure level and inherited by all substructures. Therefore the field is not explicitly included in the substructures. When a CCM composite record is built by the crsp_ccm_read_all function the primary identifier becomes the PERMNO used as the key, which is stored in the CCM_ID field of this structure. The LOADTYPE flag is set to 1 to signify that the structure is loaded with a composite record.

SECURITY LEVEL LINK DATA CONSIDERATIONS

Consider the following in order to access the new security level link data.

1. Additional security links allow multiple PERMNOs of the same company to link to the same company level data. Users must be aware that the same company data can be retrieved in multiple ways.

2. The PERMCO link is no longer needed since a secondary security can link directly between CRSP and Compustat. PERMCO can still be used to find other securities when no direct link is found.

3. Security level links are available only during the range of Compustat security data. In some cases, Compustat security data are not available as far back as company data. In others, there may be gaps of security data within a company range. CRSP fills in the available Compustat company data range so at least one link record covers all time periods in the range. If no securities are available during a range, a dummy security is generated for purposes of the link. These dummy securities always have an IID ending with X.

4. CRSP assigns a LINKPRIM marker to all link records, based on the Compustat PRIMISS marker. PRIMISS is used to identify the primary security for the company at any given time. LINKPRIM values are:

   P Primary, identified by Compustat in monthly security data.

   J Joiner secondary issue of a company, identified by Compustat in monthly security data.

   C Primary, assigned by CRSP to resolve ranges of overlapping or missing primary markers from Compustat in order to produce one primary security throughout the company history.

   N Secondary, assigned by CRSP to override Compustat. Compustat allows a US and Canadian security to both be marked as Primary at the same time. For Purposes of the link, CRSP allows only one primary at a time and marks the others as N.

5. CRSP supports an access option of primary PERMNO, or ppermno, which restricts links to only those marked primary.

6. The legacy CST format databases remain based on the old company-based links, thus using only the rows marked as primary.

CHAPTER 3: DATABASE ACCESS FUNCTIONS

ITEM OVERVIEW

CRSPAccess items in the Stock & Index databases and the CRSP\Compustat Merged database can be identified with unique mnemonic text items name maintained by CRSP, called itm_name. Items can be further qualified by a set of secondary keys. CRSP calls these known collections of keys and values a keyset and assigns a numeric code and mnemonic tag to each unique collection. Each of these represents different output series. When multiple keysets are available, the user can specify both the item and keyset to identify a series or use the default preset combination most commonly used.

For example, the Compustat data item SALE has secondary keys for industry format, data format, population source, and consolidation level. A different value of company sales can be available for any combination of these keys, such as a combination that represents the originally reported sales or the final restated sales from a later filing.

DATA ITEM GROUPING

- All itm_names are organized into groups for selection and presentation. Each group is given a name called grp_name. Grp_names are unique within the application and do not overlap with itm_name.

A group can include other groups or a group can include items. Items in a group must be compatible so that they can form a single table. For example, time series items must use the same calendar if using the same keyset. Items can belong to more than one group. Each group belongs to an entity class that describes the types of entities associated with items in the group. Company level and security level data are examples of entity classes.

Groups can be selected by grp_name, but designated groups can also be selected by a two-letter mnemonic code.

CRSP ITEM LIST SELECTION

CRSP item functions use a standard notation for specifying a set of data items. The notation allows selection by group or item. Examples are bal_ann or sale;at or prc;ret;vol or sale.*

For more information, please see Reporting Tools - ts_print section of the CRSP Utilities and Program Libraries Guide 3.90.

The item notation includes a high level item descriptor comprised of item elements, global qualifiers, and keyset specifications. The user provides information in one of three forms, with as many invocations as desired:

- **Full_list** – full description of items to select. It is in the form `[global_section:]list_section`
- **File+list** – description of items is in an input file and qualifiers
  - Specified in form `[global_section:]filepath`
  - Where file contains a `list_element` on each row.
- **Printopt** – a defined 2-letter shorthand for a defined or structure group. A printopt includes the 2-letter code optionally followed by keyset_list, separated by a period.

One list is maintained, and if an item/keyset is specified multiple times in the same handle it is ignored after the first one.

Selection components are:

- **Global_section** – optional section that modifies all elements in the list. It can contain the following markers:
  - **F** – add all children items with relation type F (footnotes)
  - **D** – add all children items with relation type D (data codes)
  - **K.keyset_list** – apply the keyset list to all items. See below for form of a keyset_list.
- List_section – semi-colon-delimited string of list elements: `list_element[;List_element…]`
- List_element – describes a CRSP item name or group name and keysets applied to it. The element name and keyset, if provided, are delimited by a period. A list element is in the form `elem_name[.keyset_list]`

- Elem_name – can be a CRSP item name or group name. Eventually all groups will be expanded to one or more items.
- Keyset_list – describes a set of keysets used for items. A keyset list is in the form
  - * - select all available keysets
  - #-#,#... – select all indicated keysets in a numeric list (examples 3 or 1-2 or 1,6-7,12 or 3-5,9-11)
  - [empty] – use the default keyset for all selected items.

ITEM HANDLE

All item handling activities for one application are stored in an item handle structure. This handle is passed as a parameter to most item handling functions. The structure contains the following types of information:

- A group table – an array of all groups available in the application. Each group includes information specific to that table –
  - An item list of all items and keysets selected for that table. All selected items are attached to one of the group tables.
  - A list of all keysets selected for that table
  - A list of structures and keysets if the group is a structure
- Set information, with database information including the set structure. The set structure is not accessed directly. The items are bound to the proper location in the set structure.
- Calendar information including the data needed to identify periods for time series data.
- Key information for the current data loaded.
- Indexes directly to the items without the group context.

All item lists use a common CRSP_ITM structure for an item. It is identified by itm_name and keyset and contains pointers to the data objects plus pointers to relevant information in the reference arrays.

CRSPAccess provides a high level and flexible programming mechanism to support the large and changing set of supported data items. These item handling functions organize all items into groups (tables) based on an application identifier tied to a defined set of data using a standard notation and data structure. Users can specify any number of items with a standard mutation and traverse these items by group or look up individual fields. Item handling will work with any database with reference data. The CRSPAccess Programmers Guide contains functions for the CRSP Stock and Index data.

ITEM FUNCTIONS

The structure of a program using the item handling functions takes the baskc form:

- Declare a `CRSP_ITM_HNDL` pointer for an `app_id` and initialize to NULL.
- Call the `crsp_itm_init` function to initialize the handle given the `app_id` and the database path. This will load all the reference data and open a shell of the database.
- Call the item load functions one or more times to select items of interest. `Crsp_itm_load` should be sufficient for most usage. This will prepare the database and attach each item to its data location.
- There are five handle configuration flags that can be set to affect the behavior or the item handling. Each has a default value set during the initialization, but can be modified by setting a field in the handle.
  - `Keytype` – determines the keys used to select data in read functions. Supported keytypes for the application are included in the reference data. A default will be set.
  - `Keyset_disp_cd` – determined whether keyset items are labeled by the keyset number (NUM), the keyset tag (TAG), or the expanded list of all items comprising the keyset (EXP). The default display is TAG.
  - `Fiscal_disp_cd` – determines whether fiscal-based time series items are reported on a calendar basis (C) or a fiscal basis (F). The default it C.
  - `Curr_disp_cd` – determines whether monetary values are reported in the currency reported by Compustat (REP) or in

US Dollars (USD). The default currency display code is REP.

- `Grp_fill_cd` – determines whether group item lists are filled so that every selected item is included for every selected keyset (Y or N). The default is Yes (Y).

- Call a function to load the key data into the handle and call `crsp_itm_get` as many times as needed to read desired data.
  - User can use `crsp_itm_set_key` to set individual input key items of interest or `crsp_itm_parse_key` to pass a string and have it parsed into one or more input key items.
  - The `crsp_itm_read` function will load all data according to items loaded and key information specified. The user can retrieve the value for any of the found keys after the read function with `crsp_itm_get_key`.
- Processing of data is usually done after the appropriate `get` is excecuted and data are loaded. A user can process data by traversing the groups and the item lists, or use one of the item indexes created to find a specific item and process it. Since a `get` function can affect either master or detail data, the program must rely on the status to determine whether a class of data was changed by the get and then process it accordingly.
- Call the `crsp_itm_close` function to close the handle and free its data.

ITEM USAGE

Normal data processing access is through an item structure named `CRSP_ITM`. `CRSP_ITM` includes metadata describing the item as well as the actual data retrieved from the database. The metadata is set once by `crsp_itm_open`, but the data that is loaded can change each time the `crsp_itm_read` function is called.

The item handling functions hide the internal storage and report the data according to itm_name and keyset, which can be found from among the loaded items with the item look-up function, `crsp_itm_find`.

CCM STRUCTURES

Selected Xpressfeed primary data groups and CRSP supplemental data groups are accessed by the entire group as a defined structure rather than as a stand-alone item. These data groups and their elements can both be accessed by itm_name, but recommended programming access is through the itm_name of the structure. To access the structure and its fields, load the structure itm_name during initialization, create a CRSP_ITM pointer matching the itm_name and attach it to the data, anc access the structure and its fields through the pointer. The tables below show the data groups available as structures and their usage through the CRSP_ITM structure.

Itm_name	Description	C STRUCTURE NAME	Object Type	Object Access via CRSP_ITM
MASTER	CCM company id and range data	CRSP_CCM_MASTER	row	master_itm->obj.row
COMPANY	CCM company header information	CRSP_CCM_COMPANY	row	company_itm->obj.row
IDX_INDEX	CCM idx_index header information	CRSP_CCM_IDX_INDEX	row	idx_index_itm->obj.row
SPIND	S&P index header (pre-GICS)	CRSP_CCM_SPIND	row	spind.itm->obj.row
COMPHIST	CCM company header history	CRSP_CCM_COMPHIST	array	comphist_itm->obj.arr
CSTHIST	CST header history	CRSP_CST_NAME	array	csthist_itm->obj.arr
LINK	link history	CRSP_CCM_LINK	array	link_itm->obj.arr
LINKUSED	CCM company CRSP link used data	CRSP_CCM_LINKUSED	array	linkused_itm->obj.arr
LINKRNG	CCM company CRSP link range data	CRSP_CCM_LINKRNG	array	linkused_itm->obj.arr
ADJFACT	CCM company adjustment factor history	CRSP_CCM_ADJFACT	array	adjfact.itm->obj.arr
HGIC	CCM company GICS code history	CRSP_CCM_HGIC	array	hgic_itm->obj.arr
OFFTITL	CCM company officer title data	CRSP_CCM_OFFTITL	array	offtitl_itm->obj.arr
CCM_FILEDATE	CCM company filing date data	CRSP_CCM_FILEDATE	array	ccm_filedate->obj.arr
CCM_IPCD	CCM industry presentation code data	CRSP_CCM_IPCD	array	ccm_filedate->obj.arr
SECURITY	CCM security header information	CRSP_CCM_SECURITY	row	security_itm->obj.row

Itm_name	Description	C STRUCTURE NAME	Object Type	Object Access via CRSP_ ITM
SECHIST	CCM security header history	CRSP_CCM_SECHIST	array	sechist_itm->obj.arr
SEC_MTHSPT	CCM security monthly split events	CRSP_CCM_SEC_ MTHSPT	row	sec_mthspt_itm->obj.row
SEC_MSPT_FN	CCM security monthly split event footnotes	CRSP_CCM_SEC_ MTH_FN	row	sec_mspt_fn_itm->obj. row
SEC_MDIV_FN	CCM security monthly dividend event footnotes	CRSP_CCM_SEC_ MTH_FN	row	sec_mdiv_fn_itm->obj.row
SEC_SPIND	CCM security S&P information events	CRSP_CCM_SEC_ SPIND	row	sec_spind_itm->obj.row
IDXCST_HIS	CCM security historical index constituents	CRSP_CCM_IDXCST_ HIS	array	idxcst_his_itm.obj.arr
CCM_SEGCUR	CCM opseg currency rate data	CRSP_CST_SEGCUR	array	ccm_segcur->obj.arr
CCM_SEGSRC	CCM opseg source data	CRSP_CST_SEGSRC	array	ccm_segsrc->obj.arr
CCM_SEGPROD	CCM opseg product data	CRSP_CST_SEGPROD	array	ccm_segprod->obj.arr
CCM_SEGCUST	CCM opseg customer data	CRSP_CST_SEGCUST	array	ccm_segcust->obj.arr
CCM_SEGDTL	CCM opseg detail data	CRSP_CST_SEGDTL	array	ccm_segdtl->obj.arr
CCM_SEGITM	CCM opseg item data	CRSP_CST_SEGITM	array	ccm_segitm->obj.arr
CCM_SEGNAICS	CCM opseg NAICS data	CRSP_CST_SEGNAICS	array	ccm_segnaics->obj.arr
CCM_SEGGEO	CCM opseg geographic data	CRSP_CST_SEGGEO	array	ccm_seggeo->obj.arr

CCM FIELD USAGE TABLE

STRUCTURE: MASTER

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
BEGQTR	Quarterly date of earliest data (yyyy.q)	int(4)	%6d	master_itm->arr.master->begqtr
BEGYR	Annual date of earlist data (yyyymmdd)	int(4)	%4d	master_itm->arr.master->begyr
CBEGDT	First date of Compustat data	int(4)	%8d	master_itm->arr.master->cbegdt
CCMID	Permanent record identifier for Compustat company or index data, represents GVKEY for company, GVKEYX for index	int(4)	%6d	master_itm->arr.master->ccmid
CCMIDTYPE	Type of key for Compustat data. 1 = company data, 2 = index data	int(4)	%2d	master_itm->arr.master- >ccmidtype
CENDT	Last date of Compustat data	int(4)	%8d	master_itm->arr.master->cendt
ENDQTR	Quarterly date of last data (yyyy.q)	int(4)	%6d	master_itm->arr.master->endqtr
ENDYR	Annual date of last data (yyyymmdd)	int(4)	%4d	master_itm->arr.master->endyr

STRUCTURE: COMPANY

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
ADD1-4	Address lines 1-4	char(68)	%-65s	company_itm->arr.company->add#
ADDZIP	Postal code	char(24)	%-24s	company_itm->arr.company->addzip
BUSDESC	Business description	char(2000)	%-2000s	company_itm->arr.company->busdesc
CIK	CIK number	char(12)	%-10s	company_itm->arr.company->cik
CITY	City	char(104)	%-104s	company_itm->arr.company->city

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
CONM	Company name	char(256)	%-255s	company_itm->arr.company->conm
CONML	Company legal name	char(104)	%-100s	company_itm->arr.company->conml
COSTAT	Postal code	char(24)	%-1s	company_itm->arr.company->addzip
COUNTY	County code	char(104)	%-100s	company_itm->arr.company->county
DLDTE	Research company deletion date	int(4)	%8d	company_itm->arr.company->dldte
DLRSN	Research company reason for deletion	char(12)	%-8s	company_itm->arr.company->dlrsn
EIN	Employer identification number	char(12)	%-10s	company_itm->arr.company->ein
FAX	Fax number	char(24)	%-18s	company_itm->arr.company->fax
FIC	ISO Country code of incorporation	char(16)	%-3s	company_itm->arr.company->fic
FYRC	Fiscal year end (current)	int(4)	%2d	company_itm->arr.company->fyrc
GGROUP	GICS groups	char(12)	%-4s	company_itm->arr.company->ggroup
GIND	GICS industries	char(12)	%-6s	company_itm->arr.company->gind
GSECTOR	GICS sectors	char(12)	%-2s	company_itm->arr.company->gsector
GSUBIND	GICS sub-industries	char(12)	%-8s	company_itm->arr.company->gsubind
IDBFLAG	International/Domestic/Both indicator	char(12)	%-1s	company_itm->arr.company->idbflag
INCORP	State/Province of incorporation code	char(12)	%-8s	company_itm->arr.company->incorp
IPODATE	Company initial public offering date	int(4)	%8d	company_itm->arr.company->ipodate
LOC	ISOCountry code/ headquarters	char(4)	%-3s	company_itm->arr.company->loc
NAICS	North American Industry Classification Code	char(8)	%-6s	company_itm->arr.company->naics
PHONE	Phone number	char(24)	%-18s	company_itm->arr.company->phone
PRICAN	Primary Issue Tag - Canada	char(12)	%-8s	company_itm->arr.company->prican
PRIROW	Primary Issue Tag – rest of world	char(12)	%-8s	company_itm->arr.company->prirow
PRIUSA	Primary Issue Tag - USA	char(12)	%-8s	company_itm->arr.company->priusa
SIC	SIC code	int(4)	%4d	company_itm->arr.company->sic
STATE	State/Province	char(12)	%-8s	company_itm->arr.company->state
STKO	Stock ownership code	int(4)	%1d	company_itm->arr.company->stko
WEBURL	Website address	char(68)	%-60s	company_itm->arr.company->weburl

STRUCTURE: IDX_INDEX

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
IDX13KEY	13 character key	char(16)	%-13s	idx_index_itm->arr.idx_index->idx13key
IDXCSTFLG	Index constituent flag	char(4)	%-2s	idx_index_itm->arr.idx_index->idxcstflg
IDXSTAT	Index Status	char(2)	%-10s	idx_index_val->arr.idx_index->idxstat
INDEXCAT	Index category code	char(12)	%-10s	idx_index_itm->arr.idx_index->indexcat
INDEXGEO	Index geographical area	char(12)	%-10s	idx_index_itm->arr.idx_index->indexgeo
INDEXTYPE	Index type	char(12)	%-10s	idx_index_itm->arr.idx_index->indextype
INDEXVAL	Index value	char(12)	%-10s	idx_index_itm->arr.idx_index->indexval
SPII	S&P industry index code	int(4)	%4d	idx_index_itm->arr.idx_index->spii
SPMI	S&P major index code	int(4)	%4d	idx_index_itm->arr.idx_index->spmi
TICI	Issue trading ticker	char(12)	%-8s	idx_index_itm->arr.idx_index->tici
XCONM	Company Name (Index)	char(256)	%-255s	idx_index_itm->arr.idx_index->xconm
XINDEXID	Index ID	char(12)	%-12s	idx_index_itm->arr.idx_index->xindexid
XTIC	Ticker/trading symbol (index)	char(10)	%-10s	idx_index_itm->arr.idx_index->xtic

STRUCTURE: SPIND

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SPIID	S&P Industry ID	int(4)	%4d	spind_itm->arr.spind->spiid
SPIMID	S&P Major Index ID	int(4)	%4d	spind_itm->arr.spind->spimid
SPITIC	S&P Index ticker	char(12)	%-12s	spind_itm->arr.spind->spitic
SPIDESC	S&P Index industry description /reference	char(256)	%-256s	spind_itm->arr.spind->spidesc

STRUCTURE: COMPHIST

Comphist field usage assumes an initialized CRSP_ITM comphist_itm attached to the COMPHIST itm_name. Index I in field usage is between 0 and comphist_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
HCHGDT	Comphist description effective date	int(4)	%8d	comphist_itm->arr.comphist[i].hchget
HCHGENDDT	Comphist description last effective date	int(4)	%8d	comphist_itm->arr.comphist[i].hchgenddt
HDLDTE	Historical research company – deletion date	int(4)	%8d	comphist_itm->arr.comphist[i].hdldte
HFYRC	Historical fiscal year end month / current	int(4)	%10d	comphist_itm->arr.comphist[i].hfyrc
HIPODATE	Historical company official public offering date	int(4)	%10d	comphist_itm->arr.comphist[i].hipodate
HSIC	Historical SIC Code	int(4)	%10d	comphist_itm->arr.comphist[i].hsic
HSPCINDCD	Historical S&P Industry code	int(4)	%10d	comphist_itm->arr.comphist[i].hspcindcd
HSPCSECCD	Historical S&P Economic sector code	int(4)	%10d	comphist_itm->arr.comphist[i].hspcseccd
HSTKO	Historical stock ownership code	int(4)	%10d	comphist_itm->arr.comphist[i].hstko
HADD1…4	Historical address lines 1-4	char(68)	%-68s	comphist_itm->arr.comphist[i].haddl#
HADDZIP	Historical postal code	char(68)	%-24s	comphist_itm->arr.comphist[i].haddzip
HBUSDESC	Historical business description	char(2000)	%-2000s	comphist_itm->arr.comphist[i].hbusdesc
HCIK	Historical CIK number	char(12)	%-12s	comphist_itm->arr.comphist[i].hcik
HCITY	Historical city	char(104)	%-104s	comphist_itm->arr.comphist[i].hcity
HCONM	Historical company name	char(256)	%-256s	comphist_itm->arr.comphist[i].hconm
HCONML	Historical legal company name	char(104)	%-104s	comphist_itm->arr.comphist[i].hconml
HCOSTAT	Historical active/inactive status marker	char(4)	%-4s	comphist_itm->arr.comphist[i].hcostat
HCOUNTY	Historical county code	char(104)	%-1044s	comphist_itm->arr.comphist[i].hcounty
HDLRSN	Historical research company reason for deletion	char(12)	%-12s	comphist_itm->arr.comphist[i].hdlrsn
HEIN	Historical employer identification number	char(12)	%-12s	comphist_itm->arr.comphist[i].hein
HFAX	Historical fax number	char(16)	%-16s	comphist_itm->arr.comphist[i].hfax
HFIC	Historical ISO country code / incorporation	char(16)	%-16s	comphist_itm->arr.comphist[i].hfic
HGGROUP	Historical GICS group	char(12)	%-12s	comphist_itm->arr.comphist[i].hggroup
HGIND	Historical GICS industries	char(12)	%-12s	comphist_itm->arr.comphist[i].hgind
HGSECTOR	Historical GICS sector	char(12)	%-12s	comphist_itm->arr.comphist[i].hgsector
HGSUBIND	Historical GICS sub-industries	char(12)	%-12s	comphist_itm->arr.comphist[i].hgsubind
HIDBFLAG	Historical international, domestic, both indicator	char(12)	%-12s	comphist_itm->arr.comphist[i].hidbflag
HINCORP	Historical state/province of incorporation code	char(12)	%-12s	comphist_itm->arr.comphist[i].hincorp

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
HLOC	Historic ISO country code/ headquarters	char(4)	%-4s	comphist_itm->arr.comphist[i].hloc
HNAICS	Historical NAICS codes	char(8)	%-8s	comphist_itm->arr.comphist[i].hnaics
HPHONE	Historical phone number	char(16)	%-16s	comphist_itm->arr.comphist[i].hphone
HPRICAN	Historical primary issue tag - Cananda	char(12)	%-12s	comphist_itm->arr.comphist[i].hprican
HPRIROW	Historical primary issue tag – rest of world	char(12)	%-12s	comphist_itm->arr.comphist[i].hprirow
HPRIUSA	Historical primary issue tag - US	char(12)	%-12s	comphist_itm->arr.comphist[i].hpriusa
HSTATE	Historical state/province	char(12)	%-12s	comphist_itm->arr.comphist[i].hstate
HWEBURL	Historical website url	char(68)	%-68s	comphist_itm->arr.comphist[i].hweburl

STRUCTURE - CSTHIST

Csthist field usage assumes an initialized CRSP_ITM csthist_itm attached to the CSTHIST itm_name. Index i in field usage is between 0 and csthist_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
CST_CHGDT	CST History effective date	int(4)	%8d	csthist_itm->arr.csthist[i].cst_chgdt
CST_CHGENDDT	CST History last effective date	int(4)	%8d	csthist_itm->arr.csthist[i].cst_chgenddt
CST_DNUM	CST History industry code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_dnum
CST_FILE	CST History file identification code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_file
CST_ZLIST	CST History exchange listing and S&P Index code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_zlist
CST_STATE	CST History state identification code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_state
CST_COUNTY	CST History county identification code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_county
CST_STINC	CST History state incorporation code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_stinc
CST_FINC	CST History foreign incorporation code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_finc
CST_XREL	CST History industry index relative code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_xrel
CST_STK	CST History stock ownership code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_stk
CST_DUP	CST History duplicate file code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_dup
CST_CCNDX	CST History current Canadian index code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_ccndx
CST_GICS	CST History Global Industry Classification Standard Code	int(4)	%4d	csthist_itm->arr.csthist[i].cst_gics
CST_IPODT	CST History IPO date	int(4)	%4d	csthist_itm->arr.csthist[i].cst_ipodt
CST_FUNDF1	CST History fundamental file identification code 1	int(4)	%4d	csthist_itm->arr.csthist[i].cst_fund1
CST_FUNDF2	CST History fundamental file identification code 2	int(4)	%4d	csthist_itm->arr.csthist[i].cst_fundf2
CST_FUNDF3	CST History fundamental file identification code 3	int(4)	%4d	csthist_itm->arr.csthist[i].cst_fundf3
CST_NAICS	CST History North American Industry Classification	char(8)	%-8s	csthist_itm->arr.csthist[i].cst_naics
CST_CPSPIN	CST History primary S&P Index marker	char(4)	%-4s	csthist_itm->arr.csthist[i].cst_cpspin
CST_CSSPIN	CST History subset S&P Index marker	char(4)	%-4s	csthist_itm->arr.csthist[i].cst_csspin
CST_CSSPII	CST History secondary S&P Index marker	char(4)	%-4s	csthist_itm->arr.csthist[i].cst_csspii
CST_SUBDBT	CST History current S&P subordinated debt rating	char(8)	%-8s	csthist_itm->arr.csthist[i].cst_subdbt

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
CST_CPAPER	CST History current S&P commercial paper rating	char(4)	%-4s	csthist_itm->arr.csthist[i].cst_cpaper
CST_SDBT	CST History current S&P senior debt rating	char(4)	%-4s	csthist_itm->arr.csthist[i].cst_sdbt
CST_SDBTIM	CST History current S&P senior debt rating - footnote	char(4)	%-4s	csthist_itm->arr.csthist[i].cst_sdbtim
CST_CNUM	CST History CUSIP issuer code	char(16)	%-16s	csthist_itm->arr.csthist[i].cst_cnum
CST_CIC	CST History issuer number	char(4)	%-4s	csthist_itm->arr.csthist[i].cst_cic
CST_CONAME	CST History company name	char(64)	%-64s	csthist_itm->arr.csthist[i].cst_coname
CST_INAME	CST History industry name	char(4)	%-4s	csthist_itm->arr.csthist[i].cst_iname
CST_SMBL	CST History stock ticker symbol	char(16)	%-16s	csthist_itm->arr.csthist[i].cst_smbl
CST_EIN	CST History employer identification number	char(16)	%-16s	csthist_itm->arr.csthist[i].cst_ein
CST_INCORP	CST History incorporation ISO country code	char(4)	%-4s	csthist_itm->arr.csthist[i].cst_incorp

STRUCTURE: LINK

Link field usage assumes an initialized CRSP_ITM link_itm attached to the LINK itm_name. Index i in field usage is between 0 and link_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
LINKDT	Effective date of the link record	int(4)	%8d	link_itm->arr.link[i].linkdt
LINKENDDT	Last effective date of the link record	int(4)	%8d	link_itm->arr.link[i].linkenddt
LPERMNO	CRSP PERMNO link during link period	int(4)	%6d	link_itm->arr.link[i].lpermno
LPERMCO	CRSP PERMCO link during link period	int(4)	%10d	link_itm->arr.link[i].lpermco
LIID	Security identifier	char(4)	%-3s	link_itm->arr.link[i].liid
LNKTYPE	Link type code	char(4)	%-4s	link_itm->arr.link[i].lnktype
LINKPRIM	Primary security link marker	char(4)	%-1s	link_itm->arr.link[i].linkprim

STRUCTURE: LINKUSED

Linkused field usage assumes an initialized CRSP_ITM linkused_itm attached to the LINKUSED itm_name. Index i in field usage is between 0 and linkused_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
ULINKDT	Effective date of the link	int(4)	%8d	linkused_itm->arr.linkused[i].ulinkdt
ULINKENDDT	Last effective date of the link	int(4)	%8d	linkused_itm->arr.linkused[i].ulinkenddt
ULINKID	Linkused row identifier	int(4)	%d	linkused_itm->arr.linkused[i].ulinkid
UGVKEY	GVKEY used in the link	int(4)	%6d	linkused_itm->arr.linkused[i].ugvkey
UPERMNO	CRSP PERMNO link during link period	int(4)	%6d	linkused_itm->arr.linkused[i].upermno
UPERMCO	CRSP PERMCO link during link period	int(4)	%6d	linkused_itm->arr.linkused[i].upermco
UIID	Used Security ID	char(4)	%-3s	linkused_itm->arr.linkused[i].uiid
USEDFLAG	Flag marking whether link is used in building composite record	char(4)	%d	linkused_itm->arr.linkused[i].usedflag
ULINKPRIM	Used link primary marker	char(4)	%-1s	linkused_itm->arr.linkused[i].ulinkprim
ULINKTYPE	Used link type	char(4)	%-4s	linkused_itm->arr.linkused[i].ulinktype

STRUCTURE: LINKRNG

Linkrng field usage assumes an initialized CRSP_ITM linkrng_itm attached to the LINKRNG itm_name. Index i in field usage is between 0 and linkrng_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
RLINKID	Linkused row identifier	int(4)	%8d	linkrng_itm->arr.linkrng[i].rlinkid
RKEYSET	Keyset applicable to range	int(4)	%8d	linkrng_itm->arr.linkrng[i].rkeyset
RCALID	Calendar applicable to range	int(4)	%8d	linkrng_itm->arr.linkrng[i].rcalid
RBEGIND	Beginning time series range of link	int(4)	%8d	linkrng_itm->arr.linkrng[i].rbegind
RENDIND	Ending time series range of link	int(4)	%8d	linkrng_itm->arr.linkrng[i].rendind
RPREVIND	Time series range immediately preceding the link	int(4)	%8d	linkrng_itm->arr.linkrng[i].rprevind
RBEGDT	Beginning calendar range of link	int(4)	%8d	linkrng_itm->arr.linkrng[i].rbegdt
RENDDT	Ending calendar range of link	int(4)	%8d	linkrng_itm->arr.linkrng[i].renddt
RPREVDT	Ending calendar range preceding the link	int(4)	%8d	linkrng_itm->arr.linkrng[i].rprevdt
RFISCAL_ DATA_FLG	Type of time series, C-calendar or F-fiscal.	char(8)	%-8s	linkrng_itm->arr.linkrng[i].rfiscal_ data_flg

STRUCTURE: ADJFACT

Adjfact field usage assumes an initialized CRSP_ITM adjfact_itm attached to the ADJFACT itm_name. Index i in field usage is between 0 and adjfact_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
EFFDATE	Effective date- company cumulative factor	int(4)	%10d	linkrng_itm->arr.linkrng[i].effdate
THRUDATE	Thu date – company cumulative factor	int(4)	%10d	linkrng_itm->arr.linkrng[i].thrudate
ADJEX	Cumulative adjustment factor by Ex-date	double(8)	%18.4f	adjfact_itm->arr.adjfact[i].adjex
ADJPAY	Cumulative adjustment factor by Pay-date	double(8)	%18.4f	adjfact_itm->arr.adjfact[i].adjpay

STRUCTURE: HGIC

Hgic field usage assumes an initialized CRSP_ITM hgic_itm attached to the HGIC itm_name. Index i in field usage is between 0 and hgic_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
INDFROM	Effective from (start) date	int(4)	%8d	hgic_itm->arr.hgic[i].indfrom
INDTHRU	Effective through (last) date	int(4)	%8d	hgic_itm->arr.hgic[i].indthru
GGROUPH	Industry group name	char(12)	%-12s	hgic_itm->arr.hgic[i].ggrouph
GINDH	Group industry	char(12)	%-12s	hgic_itm->arr.hgic[i].gindh
GSECTORH	Group industry sector	char(12)	%-12s	hgic_itm->arr.hgic[i].gsectorh
GSUBINDH	Group sub-industries	char(12)	%-12s	hgic_itm->arr.hgic[i].gsubindh

STRUCTURE: OFFTITL

Offtitl field usage assumes an initialized CRSP_ITM offtitl_itm attached to the OFFTITL itm_name. Index i in field usage is between 0 and offtitl_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
OFID	Officer ID	int(4)	%9d	offtitl_itm->arr.offtitl[i].ofid
OFCD	Officer title	char(16)	%-8s	offtitl_itm->arr.offtitl[i].ofcd
OFNM	Officer Name(s)	char(40)	%-39s	offtitl_itm->arr.offtitl[i].ofnm

STRUCTURE: CCM_FILEDATE

Ccm_filedate field usage assumes an initialized CRSP_ITM ccm_filedate_itm attached to the CCM_FILEDATE itm_name. Index i in field usage is between 0 and ccm_filedate_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
FDATADATE	Company filing data date	int(4)	%8d	ccm_filedate_itm->arr.ccm_filedate[i].fdatadate
FCONSOL	Company consolidation level filedate	char(2)	%-2s	ccm_filedate_itm->arr.ccm_filedate[i].fconsol
FPOPSRC	Population source filedate	char(2)	%-2s	ccm_filedate_itm->arr.ccm_filedate[i].fpopsrc
SRCTYPE	Document source type filedate	char(12)	%-12s	ccm_filedate_itm->arr.ccm_filedate[i].srctype
FILEDATE	Company filing date	int(4)	%8d	ccm_filedate_itm->arr.ccm_filedate[i].filedate

STRUCTURE: CCM_IPCD

Ccm_ipcd field usage assumes an initialized CRSP_ITM ccm_ipcd_itm attached to the CCM_IPCD itm_name. Index i in field usage is between 0 and ccm_ipcd_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
IPDATADATE	Industry presentation code data date	int(4)	%8d	ccm_ipcd_itm->arr.ccm_ipcd[i].ipdatadate
IPCONSOL	Level of consolidation (Industry presentation code)	char(2)	%-1s	ccm_ipcd_itm->arr.ccm_ipcd[i].ipconsol
IPPOPSRC	Population source (Industry presentation code)	char(2)	%-1s	ccm_ipcd_itm->arr.ccm_ipcd[i].ippopsrc
IPCD	Industry presentation code	char(2)	%-1s	ccm_ipcd_itm->arr.ccm_ipcd[i].ipcd

STRUCTURE: SECURITY

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
EXCHG	Stock exchange	int(4)	%4d	security_itm->arr.security->exchg
DLDTEI	Security inactivation date	int(4)	%8d	security_itm->arr.security->dltei
IID_SEQ_ NUM	IID sequence number	int(4)	%8d	security_itm->arr.security->iid_seq_num
SBEGDT	First date of Compustat data for issue	int(4)	%8d	security_itm->arr.security->sbegdt
SENDDT	Last date of Compustat data for issue	int(4)	%8d	security_itm->arr.security->senddt
IID	Issue ID	char(4)	%-3s	security_itm->arr.security->iid
SCUSIP	CUSIP	char(12)	%-12s	security_itm->arr.security->cusip
DLRSNI	Security inactivation code	char(12)	%-8s	security_itm->arr.security->dlrsni
DSCI	Security description	char(32)	%-28s	security_itm->arr.security->dsci
EPF	Earnings participation flag	char(4)	%-1s	security_itm->arr.security->epf

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
EXCNTRY	Stock exchange country code	char(4)	%-3s	security_itm->arr.security->excntry
ISIN	International security identification number	char(16)	%-12s	security_itm->arr.security->isin
SSECSTAT	Security status marker	char(4)	%-4s	security_itm->arr.security->ssecstat
SEDOL	SEDOL	char(8)	%-7s	security_itm->arr.security->sedol
TIC	Ticker/trading symbol	char(12)	%-8s	security_itm->arr.security->tic
TPCI	Issue type	char(12)	%-8s	security_itm->arr.security->tpci

STRUCTURE: SECHIST

Sechist field usage assumes an initialized CRSP_ITM sechist_itm attached to the SECHIST itm_name. Index i in field usage is between 0 and sechist_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
HSCHGDT	Historical security change date	int(4)	%8d	sechist_itm->arr.sechist[i].hschgdt
HSCHGENDDT	Historical security change end date	int(4)	%8d	sechist_itm->arr.sechist[i].hschgenddt
HEXCHG	Historical stock exchange	int(4)	%10d	sechist_itm->arr.sechist[i].hexchg
HDLDTEI	Historical security inactivation date	int(4)	%8d	sechist_itm->arr.sechist[i].hdldtei
HIID_SEQ_NUM	Historical issue ID sequence number	int(4)	%10d	sechist_itm->arr.sechist[i].hiid_seq_num
HIID	Historical issue ID	char(4)	%-3s	sechist_itm->arr.sechist[i].hiid
HSCUSIP	Historical CUSIP	char(12)	%-12s	sechist_itm->arr.sechist[i].hscusip
HDLRSNI	Historical security inactivation code	char(12)	%-12s	sechist_itm->arr.sechist[i].hdhdrsni
HDSCI	Historical security description	char(32)	%-32s	sechist_itm->arr.sechist[i].hdsci
HEPF	Historical earnings participation flag	char(4)	%-4s	sechist_itm->arr.sechist[i].hepf
HEXCNTRY	Historical stock exchange country code	char(4)	%-4s	sechist_itm->arr.sechist[i].hexcntry
HISIN	Historical international security identification number	char(16)	%-16s	sechist_itm->arr.sechist[i].hisin
HSSECSTAT	Historical security status marker	char(4)	%-4s	sechist_itm->arr.sechist[i].hssecstat
HSEDOL	Historical SEDOL	char(8)	%-8s	sechist_itm->arr.sechist[i]hsedol.
HTIC	Historical ticker/trading symbol	char(12)	%-12s	sechist_itm->arr.sechist[i].htic
HTPCI	Historical issue type	char(12)	%-12s	sechist_itm->arr.sechist[i].htpci

STRUCTURE: SEC_MTHSPT

Sec_mthspt field usage assumes an initialized CRSP_ITM sec_mthspt_itm attached to the MTHSPT itm_name. Index i in field usage is between 0 and mthspt_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
DATADATEM	Monthly adjustment factor data date	int(4)	%10d	sec_mthspt_itm->arr.sec_mthspt[i].datadatem
RAWPM	Raw adjustment factor – pay date - monthly	double(8)	%18.4lf	sec_mthspt_itm->arr.sec_mthspt[i].rawpm
RAWXM	Raw adjustment factor – ex date - monthly	double(8)	%18.4lf	sec_mthspt_itm->arr.sec_mthspt[i].rawxm

STRUCTURE: SEC_MSPT_FN

Sec_mspt_fn field usage assumes an initialized CRSP_ITM sec_smpt_fn_itm attached to the SEC_MSPT_FN itm_name. Index i in field usage is between 0 and sec_mspt_fn_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
DATADATEMF	Monthly adjustment factor footnote data date	int(4)	%10d	sec_mspt_fn_itm->arr.sec_mspt_fn[i].datadatemf
DATAITEMMF	Monthly split footnote dataitem	char(8)	%-8s	sec_mspt_fn_itm->arr.sec_mspt_fn[i].dataitemmf
RAWPM_FN1..FN5	Raw adjustment factor – pay date – monthly – footnotes 1-5	char(4)	%-4s	sec_mspt_fn_itm->arr.sec_mspt_fn[i].rawpm_fn1..fn5
RAWXM_FN1..FN5	Raw adjustment factor – ex date – monthly – footnotes 1-5	char(4)	%-4s	sec_mspt_fn_itm->arr.sec_mspt_fn[i].rawxm_fn1..fn5

STRUCTURE: SEC_MDIV_FN

Sec_mdiv_fn field usage assumes an initialized CRSP_ITM sec_mdiv_fn_itm attached to the SEC_MDIV_FN itm_name. Index i in field usage is between 0 and sec_mdiv_fn_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
DIVDATADATEMF	Monthly dividend footnote data date	int(4)	%10d	sec_mdiv_fn_itm->arr.sec_mdiv_fn[i].divdatadatemf
DIVDATAITEMMF	Monthly dividend footnote data item	char(8)	%-8s	sec_mdiv_fn_itm->arr.sec_mdiv_fn[i].divdataitemmf
DVPSPM_FN1..FN5	Dividend per share – pay date – monthly – footnotes 1-5	char(4)	%-4s	sec_mdiv_fn_itm->arr.sec_mdiv_fn[i].dvpspm_fn1..fn5
DVPSXM_FN1..FN5	Dividend per share – ex date – monthly – footnotes 1-5	char(4)	%-4s	sec_mdiv_fn_itm->arr.sec_mdiv_fn[i].dvpsxm_fn1..fn5

STRUCTURE: SEC_SPIND

Sec_spind field usage assumes an initialized CRSP_ITM sec_spind_itm attached to the SEC_SPIND itm_name. Index i in field usage is between 0 and sec_spind_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SPBEGDATE	S&P Index event beginning date	int(4)	%10d	sec_spind_itm->arr.sec_spind[i].spbegdate
SPENDDATE	S&P Index event ending date	int(4)	%10d	sec_spind->arr.sec_spind[i].spenddate
SPHIID	S&P holdings industry index ID	int(4)	%4d	sec_spind_itm->arr.sec_spind[i].sphiid
SPHMID	S&P holdings major index ID	int(4)	%4d	sec_spind->arr.sec_spind[i].sphmid
SPHSEC	S&P holdings sector code	int(4)	%4d	sec_spind_itm->arr.sec_spind[i].sphsec
SPH100	S&P holdings S&P 100 marker	int(4)	%4d	sec_spind->arr.sec_spind[i].sph100
SPHCUSIP	S&P holdings CUSIP	char(12)	%-9s	sec_spind_itm->arr.sec_spind[i].sphcusip
SPHNAME	S&P holdings name	char(36)	%-31s	sec_spind->arr.sec_spind[i].sphname
SPHTIC	S&P holdings ticker	char(12)	%-8s	sec_spind_itm->arr.sec_spind[i].sphtic
SPHVG	S&P holdings value/growth indicator	char(4)	%-1s	sec_spind->arr.sec_spind[i].sphvg

STRUCTURE: IDXCST_HIS

Idxcst_his field usage assumes an initialized CRSP_ITM idx_csthis_itm attached to the IDX_CSTHIS itm_name. Index i in field usage is between 0 and idx_csthis_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
XFROM	S&P constituent from event date	int(4)	%10d	idx_csthis_itm->arr.idx_csthis[i].xfrom
IDX13KEY	S&P 13 character key	char(16)	%-13s	Idx_csthis->arr.idx_csthis[i].idx13key
XGVKETX	S&P constituent event index GVKEY	int(4)	%10d	Idx_csthisitm->arr.idx_csthis[i].xgvkeyx

STRUCTURE: SPIDX_CST

Spidx_cst field usage assumes an initialized CRSP_ITM spidx_cst_itm attached to the SPIDX_CST itm_name. Index i in field usage is between 0 and spidx_cst_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SXBEGDATE	S&P constituent event beginning date	int(4)	%10d	spidx_cst_itm->arr.spidx_cst[i].sxbegdate
SXENDDATE	S&P constituent event ending date	int(4)	%10d	spidx_cst_itm->arr.spidx_cst[i].sxenddate
SPFLOAT	S&P float shares	int(4)	%4d	spidx_cst_itm->arr.spidx_cst[i].spfloat
INDEXID	S&P major index ID	char(12)	%-10s	spidx_cst_itm->arr.spidx_cst[i].indexid
EXCHGX	S&P constituent exchange	char(8)	%-4s	spidx_cst_itm->arr.spidx_cst[i].exchgx
TICX	S&P holdings ticker	char(12)	%-10s	spidx_cst_itm->arr.spidx_cst[i].ticx
CUSIPX	S&P constituent CUSIP	char(12)	%-9s	spidx_cst_itm->arr.spidx_cst[i].cusipx
CONMX	S&P constituent name	char(44)	%-40s	spidx_cst_itm->arr.spidx_cst[i].conmx
CONTYPE	S&P constituent type	char(12)	%-10s	spidx_cst_itm->arr.spidx_cst[i].contype
CONVAL	S&P constituent value	char(12)	%-10s	spidx_cst_itm->arr.spidx_cst[i].conval

STRUCTURE: CCM_SEGCUR

Ccm_segcur field usage assumes an initialized CRSP_ITM ccm_segcur_itm attached to the CCM_SEGCUR itm_name. Index i in field usage is between 0 and ccm_segcur_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SC_DATAYR	Data year	int(4)	%4d	ccm_segcur_itm->arr.ccm_segcur[i].sc_datayr
SC_DATAFYR	Data fiscal year end month	int(4)	%2d	ccm_segcur_itm->arr.ccm_segcur[i].sc_datafyr
SC_CALYR	Data calendar year	int(4)	%4d	ccm_segcur_itm->arr.ccm_segcur[i].sc_calyr
SC_SRCYFYR	Source fiscal year end month	int(4)	%2d	ccm_segcur_itm->arr.ccm_segcur[i].sc_srcfyr
SC_XRATE	Period end exchange rate	double(8)	%16.8f	ccm_segcur_itm->arr.ccm_segcur[i].sc_xrate
SC_XRATE12	12-month moving exchange rate	double(8)	%16.8f	ccm_segcur_itm->arr.ccm_segcur[i].sc_xrate12
SC_SRCCUR	Source currency code	char(4)	%-3s	ccm_segcur_itm->arr.ccm_segcur[i].sc_srccur
SC_CURCD	ISO currency code (USD)	char(4)	%-3s	ccm_segcur_itm->arr.ccm_segcur[i].sc_curcd

STRUCTURE: CCM_SEGSRC

Ccm_segsrc field usage assumes an initialized CRSP_ITM ccm_segsrc_itm attached to the CCM_SEGSRC itm_name. Index i in field usage is between 0 and ccm_segsrc_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SS_SRCYR	Source year	int(4)	%4d	ccm_segsrc_itm->arr.ccm_segsrc[i].ss_srcyr
SS_SRCFYR	Source fiscal year end month	int(4)	%2d	ccm_segsrc_itm->arr.ccm_segsrc[i].ss_srcfyr
SS_CALYR	Data calendar year	int(4)	%4d	ccm_segsrc_itm->arr.ccm_segsrc[i].ss_calyr
SS_RCST1	Reserved 1	int(4)	%4d	ccm_segsrc_itm->arr.ccm_segsrc[i].ss_rcst1
SS_SSRCE	Source document code	char(4)	%-2s	ccm_segsrc_itm->arr.ccm_segsrc[i].ss_ssrce
SS_SUCODE	Source update code	char(4)	%-2s	ccm_segsrc_itm->arr.ccm_segsrc[i].ss_sucode
SS_CURCD	ISO currency code	char(4)	%-3s	ccm_segsrc_itm->arr.ccm_segsrc[i].ss_curcd
SS_SRCCUR	Source ISO currency code	char(4)	%-3s	ccm_segsrc_itm->arr.ccm_segsrc[i].ss_srccur
SS_HNAICS	Segment primary historical NAICS	char(8)	%-6s	ccm_segsrc_itm->arr.ccm_segsrc[i].ss_hnaics

STRUCTURE: CCM_SEGPROD

Ccm_segprod field usage assumes an initialized CRSP_ITM ccm_segprod_itm attached to the CCM_SEGPROD itm_name. Index i in field usage is between 0 and ccm_segprod_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SP_SRCYR	Source year	int(4)	%4d	ccm_segprod_itm->arr.ccm_segprod[i].sp_srcyr
SP_SRCFYR	Source fiscal year end month	int(4)	%2d	ccm_segprod_itm->arr.ccm_segprod[i].sp_srcfyr
SP_PDID	Product identifier	int(4)	%4d	ccm_segprod_itm->arr.ccm_segprod[i].sp_pdid
SP_PSID	Segment link – segment identifier	int(4)	%4d	ccm_segprod_itm->arr.ccm_segprod[i].sp_psid
SP_PSALE	External revenues	float(4)	%10.4f	ccm_segprod_itm->arr.ccm_segprod[i].sp_psale
SP_RCST1	Reserved 1	float(4)	%10.4f	ccm_segprod_itm->arr.ccm_segprod[i].sp_rcst1
SP_PNAICS	Product NAICS code	char(8)	%-6s	ccm_segprod_itm->arr.ccm_segprod[i].sp_pnaics
SP_PSTYPE	Segment link- segment type	char(16)	%-83s	ccm_segprod_itm->arr.ccm_segprod[i].sp_pstype
SP_PNAME	Product name	char(64)	%-64.64s	ccm_segprod_itm->arr.ccm_segprod[i].sp_pname

STRUCTURE: CCM_SEGCUST

Ccm_segcust field usage assumes an initialized CRSP_ITM ccm_segcust_itm attached to the CCM_SEGCUST itm_name. Index i in field usage is between 0 and ccm_segcust_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SC_SRCYR	Source year	int(4)	%4d	ccm_segcust_itm->arr.ccm_segcust[i].sc_srcyr
SC_SRCFYR	Source fiscal year end month	int(4)	%2d	ccm_segcust_itm->arr.ccm_segcust[i].sc_srcfyr
SC_CDID	Customer identifier	int(4)	%4d	ccm_segcust_itm->arr.ccm_segcust[i].sc_cdid
SC_CSID	Segment link – segment identifier	int(4)	%4d	ccm_segcust_itm->arr.ccm_segcust[i].sc_csid
SC_CSALE	Customer revenues	float(4)	%10.4f	ccm_segcust_itm->arr.ccm_segcust[i].sc_csale
SC_RCST1	Reserved 1	int(4)	%4d	ccm_segcust_itm->arr.ccm_segcust[i].sc_rcst1
SC_CTYPE	Customer type	char(16)	%-8s	ccm_segcust_itm->arr.ccm_segcust[i].sc_ctype
SC_CGEOCD	Geographic area code	char(16)	%-8s	ccm_segcust_itm->arr.ccm_segcust[i].sc_cgeocd
SC_CGEOAR	Geographic area type	char(16)	%-8s	ccm_segcust_itm->arr.ccm_segcust[i].sc_cgeoar

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SC_CSTYPE	Segment link – segment type	char(16)	%-8s	ccm_segcust_itm->arr.ccm_segcust[i].sc_cstype
SC_CNAME	Customer name data	char(64)	%-64.64s	ccm_segcust_itm->arr.ccm_segcust[i].sc_cname

STRUCTURE: CCM_SEGDTL

Ccm_segdtl field usage assumes an initialized CRSP_ITM ccm_segdtl_itm attached to the CCM_SEGDTL itm_name. Index i in field usage is between 0 and ccm_segdtl_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SD_SRCYR	Source year	int(4)	%4d	ccm_segdtl_itm->arr.ccm_segdtl[i].sd_srcyr
SD_SRCFYR	Source fiscal year end month	int(4)	%2d	ccm_segdtl_itm->arr.ccm_segdtl[i].sd_srcfyr
SD_SID	Segment identifier	int(4)	%4d	ccm_segdtl_itm->arr.ccm_segdtl[i].sd_sid
SD_RCST1	Reserved 1	int(4)	%4d	ccm_segdtl_itm->arr.ccm_segdtl[i].sd_rcst1
SD_STYPE	Segment type	char(16)	%-8s	ccm_segdtl_itm->arr.ccm_segdtl[i].sd_stype
SD_SOPTP1	Operating segment type 1	char(16)	%-8s	ccm_segdtl_itm->arr.ccm_segdtl[i].sd_ctype
SD_SOPTP2	Operating segment type	char(16)	%-8s	ccm_segdtl_itm->arr.ccm_segdtl[i].sd_cgeocd
SD_SGEOTP	Geographic segment type	char(16)	%-8s	ccm_segdtl_itm->arr.ccm_segdtl[i].sd_cgeoar
SD_SNAME	Segment name	char(256)	%-64.64s	ccm_segdtl_itm->arr.ccm_segdtl[i].sd_cname

STRUCTURE: CCM_SEGITM

Ccm_segitm field usage assumes an initialized CRSP_ITM ccm_segitm_itm attached to the CCM_SEGITM itm_name. Index i in field usage is between 0 and ccm_segitm_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SI_DATYR	Data year	int(4)	%4d	ccm_segitm_itm->arr.ccm_segitm[i].si_datyr
SI_FISCYR	Data fiscal year end month	int(4)	%4d	ccm_segitm_itm->arr.ccm_segitm[i].si_fiscyr
SI_SRCYR	Source year	int(4)	%4d	ccm_segitm_itm->arr.ccm_segitm[i].si_srcyr
SI_SRCFYR	Source fiscal year end month	int(4)	%2d	ccm_segitm_itm->arr.ccm_segitm[i].si_crsfyr
SI_CALYR	Data calendar year	int(4)	%4d	ccm_segitm_itm->arr.ccm_segitm[i].si_calyr
SI_SID	Segment identifier	int(4)	%4d	ccm_segitm_itm->arr.ccm_segitm[i].si_sid
SI_EMP	Employees	int(4)	%9d	ccm_segitm_itm->arr.ccm_segitm[i].si_emp
SI_SALE	Net sales	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_sale
SI_OIBD	Operating income before depreciations	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_oibd
SI_DP	Depreciation & amortization	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_dp
SI_OIAD	Operating income after depreciation	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_oiad
SI_CAPX	Capital expenditures	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_capx
SI_IAT	Identifiable assets	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_iat
SI_EQEARN	Equity in earnings	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_eqearn
SI_INVEQ	Investments at equity	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_inveq
SI_RD	Research & development	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_rd
SI_OBKLG	Order backlog	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_obklg
SI_EXPORTS	Export sales	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_exports
SI_INTSEG	Inter-segment eliminations	int(4)	%4d	ccm_segitm_itm->arr.ccm_segitm[i].si_intseg
SI_OPINC	Operating profit	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_opinc

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SI_PI	Pretax income	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_pi
SI_IB	Income before extraordinary earnings	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_ib
SI_NI	Net income (loss)	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_ni
SI_RCST1	Reserved 1	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_rcst1
SI_RCST2	Reserved 2	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_rcst2
SI_RCST3	Reserved 3	float(4)	%10.4f	ccm_segitm_itm->arr.ccm_segitm[i].si_rcst3
SI_SALEF	Footnote 1 - sales	char(16)	%-8s	ccm_segitm_itm->arr.ccm_segitm[i].si_salef
SI_OPINCF	Footnote 2 – operating profit	char(16)	%-8s	ccm_segitm_itm->arr.ccm_segitm[i].si_opincf
SI_CAPXF	Footnote 3 – capital expenditures	char(16)	%-8s	ccm_segitm_itm->arr.ccm_segitm[i].si_capxf
SI_EQEARNF	Footnote 4 – equity in earnings	char(16)	%-8s	ccm_segitm_itm->arr.ccm_segitm[i].si_eqearnf
SI_EMPF	Footnote 5 - employees	char(16)	%-8s	ccm_segitm_itm->arr.ccm_segitm[i].si_empf
SI_RDF	Footnote 6 – research & development	char(16)	%-8s	ccm_segitm_itm->arr.ccm_segitm[i].si_rdf
SI_STYPE	Segment type	char(16)	%-8s	ccm_segitm_itm->arr.ccm_segitm[i].si_stype

STRUCTURE: CCM_SEGNAICS

Ccm_segnaics field usage assumes an initialized CRSP_ITM ccm_segnaics_itm attached to the CCM_SEGNAICS itm_name. Index i in field usage is between 0 and ccm_segnaics_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SN_SRCYR	Source year	int(4)	%4d	ccm_segnaics_itm->arr.ccm_segnaics[i].sn_srcyr
SN_SRCFYR	Source fiscal year end month	int(4)	%2d	ccm_segnaics_itm->arr.ccm_segnaics[i].sn_srcfyr
SN_SID	Segment identifier	int(4)	%4d	ccm_segnaics_itm->arr.ccm_segnaics[i].sn_sid
SN_RCST1	Reserved 1	int(4)	%4d	ccm_segnaics_itm->arr.ccm_segnaics[i].sn_rcst1
SN_STYPE	Segment type	char(16)	%-8s	ccm_segnaics_itm->arr.ccm_segnaics[i].sn_stype
SN_SNAICS	NAICS code	char(8)	%-6s	ccm_segnaics_itm->arr.ccm_segnaics[i].sn_snaics
SN_RANK	Ranking	int(4)	%4d	ccm_segnaics_itm->arr.ccm_segnaics[i].sn_rank
SN_SIC	Segment SIC code	int(4)	%4d	ccm_segnaics_itm->arr.ccm_segnaics[i].sn_sic

STRUCTURE: CCM_SEGGEO

Ccm_seggeo field usage assumes an initialized CRSP_ITM ccm_seggeo_itm attached to the CCM_SEGGEO itm_name. Index i in field usage is between 0 and ccm_seggeo_itm-> obj.arr->num -1.

Mnemonic	Field Name	Internal Storage	Display Format	Field Usage
SG_SRCYR	Source year	int(4)	%4d	ccm_seggeo_itm->arr.ccm_seggeo[i].sg_srcyr
SG_SRCFYR	Source fiscal year end month	int(4)	%2d	ccm_seggeo_itm->arr.ccm_seggeo[i].sg_srcfyr
SG_SID	Segment identifier	int(4)	%4d	ccm_seggeo_itm->arr.ccm_seggeo[i].sg_sid
SG_RCST1	Reserved 1	int(4)	%4d	ccm_seggeo_itm->arr.ccm_seggeo[i].sg_rcst1
SG_STYPE	Segment type	char(16)	%-8s	ccm_seggeo_itm->arr.ccm_seggeo[i].sg_stype
SG_SGEOCD	Geographic area code	char(16)	%-8s	ccm_seggeo_itm->arr.ccm_seggeo[i].sg_sgeocd
SG_SGEOTP	Geographic area type	char(16)	%-8s	ccm_seggeo_itm->arr.ccm_seggeo[i].sg_sgeotp

ITEM ACCESS FUNCTIONS

crsp_itm_init

Prepare a handle for item handling operations

Prototype:	`int crsp_itm_init(CRSP_ITM_HNDL **hndl, char *dbpath, init app_id, char *hndl_name))`
Description:	Prepare a handle for item handling operation for one database and one application id. The handle will be initialized and the database set type and set id identified, allowing loading of reference data and allocation of a set structure.

Arguments:	`CRSP_ITM_HNDL **hndl -`
	Double pointer to the item handle that will be used to manage the database information and item lists.
	`char *dbpath -`
	Pointer to the database containing the data to load and the applicable reference data.
	`int app_id -`
	Identifier of a defined application organizing data items into groups for access. Available app_ids can be found in the reference array, crsp_itm_app. Common app_ids have defined constants:
	• `CRSP_CCMITEMS_ID = 7` (generic CCM usage application)
	`char *hndl name –`
	Name to assign to the handle.

Return Values:	`CRSP_SUCCESS`: If initialized successfully `CRSP_FAIL`: If there is an error in the parameter, database cannot be opened, reference data unavailable, incompatibility between database and app_id.
Side Effects:	If successful, the handle data are loaded: • The handle itself is lallocated if not already allocated. • The handle fields are initialized, including all lists and arrays. • The ca_ref structure is loaded with the reference data in the database. If an old database with no reference data, it will use a global reference file with a standard name based on the app_id in the CRSP_LIB directory. • Itm_grp and itm_avail arrays in the handle are loaded with available tables and items • Set_list element is allocated using the database path and setid. The database is opened with a 0 wanted, which loads reference data but allocates no module space. The root information is loaded to the set's CRSP_ROOT_INFO structure.
Preconditions:	The item handle must be initialized to NULL or point to an allocated handle that can be re-initialized. The app_id must exist in the reference data of the database opened.

crsp_itm_open

Opens databases indicated by a handle and registers selected items.

Prototype:	`int crsp_itm_open (CRSP_ITM_HNDL **hndl)`
Description:	Registers selected items in a handle by expanding structures and keysets, preparing keys, determining modules needed to access items, opens the needed modules, and binds data in the item lists to the data structure locations. It also builds a master index of all items available in the handle.
Arguments:	`CRSP_ITM_HNDL **hndl` – Pointer to the item handle containing the needed set structure information and the current item list.
Return Values:	`SUCCESS`: If opens successfully and binds the data
	`CRSP_FAIL`: If error in parameters, inconsistent handle, error opening databases or binding items.
Side Effects:	If successful, the handle is ready for access: • The itm_set element for the database will be updated with the wanted needed. • Prepare supplementary lists for table keyset and calendar items, and handle key items. • The itm_set database will be opened with the needed wanted. • The itm_idx master list of items will be built from selected items. • All items in the list will have object pointer set to the data location in the set data structure. • If the handle grp_fill_cd is "Y", then the item lists are filled to ensure full tables. Filling creates items to ensure that every itm_name and keyset present in a group each combination is present even if not specified. Filling also arranges the lists so if multiple keysets, each is sorted in the same order as the first keyset seen.
Preconditions:	The item handle must be previously initialized with crsp_itm_init. It generally follows one or more instances of item load functions.

crsp_itm_clear

Sets missing values in objects for all items in the handle.

Prototype:	`int crsp_itm_clear (CRSP_ITM_HNDL *hndl, int clear_flag)`
Description:	Sets missing values in the objects for all items in a handle. A flag determines how the missing values are set.
Arguments:	`CRSP_ITM_HNDL *hndl` – Pointer to the item handle containing the needed set structure information and the current item list. `int clear_flag` – Code determining how the object will be cleared. Possible values are: • `CRSP_CLEAR_INIT` – only reset num for CRSP_ARRAYs and beg and end for CRSP_TIMESERIES, nothing for CRSP_ROWs. • `CRSP_CLEAR_LOADED` – only set missing values in a time series or array if non-empty. • `CRSP_CLEAR_RANGE` – set missing values for elements between beg and end for CRSP_TIMESERIES, between 0 and num – 1 for CRSP_ARRAYs, and all for CRSP_ROWs. • `CRSP_CLEAR_ALL` – set range to missing and set missing values for all elements in the object arrays. • `CRSP_CLEAR_SET` – place missing values in the 0th element of a CRSP_TIMESERIES array or the maxarr-1th element of a CRSP_ARRAY to missing values specific to the array type, or all missing values in a CRSP_ROW element.
Return Values:	`CRSP_SUCCESS`: If data loaded successfully `CRSP_FAIL`: If error with parameters, inconsistent handle, or unknown object type, array type, or all missing values in a CRSP ROW element.
Side Effects:	Object pointers found in the handle will be cleared based on the clear_flag.
Preconditions:	The item handle must be previously opened and objects bound with `crsp_itm_open`.

crsp_itm_load_key

Sets the key to be used for reads.

Prototype:	`int crsp_itm_load_key(CRSP_ITM_HNDL *hndl, char *keytype)`
Description:	Defines the keytype that will be used for subsequent reads.

Arguments:	`CRSP_ITM_HNDL *hndl -`
	Pointer to the item handle containing the needed set structure information and the current item list.
	
	`char * keytype -`
	Name of the key to initialize. Values are:
	• gvkey – Compustat company key (default)
	• gvkeyx – Compustat index key
	• ccmid – gvkey or gvkeyx
	• permno – CRSP permno found in any links
	• permco – CRSP permco found in any links
	• apermno – CRSP-centric composite records by permno
	• ppermco – CRSP-centric composite records by permno – primary links only
	• sic – Compustat company SIC code
	• ticker – Compustat security ticker symbol
	• cusip – security CUSIP

Return Values:	`CRSP_SUCCESS:`
	If successful
	`CRSP_FAIL:`
	Error in parameters, handle not initialized, or keytype not found.

Side Effects:	If successful, the handle is prepared to handle reads.
Preconditions:	The item handle must be initialized. Keytype must be known for the app_id.

crsp_itm_set_key

Sets the key specifications to be used with selecting data.

Prototype:	`int crsp_itm_set_key (CRSP_ITM_HNDL *hndl, char *key_itm, void *keyval)`
Description:	Loads key information that will be used to load data in a `crsp_itm_read` call. The key is setup during the `crsp_itm_open` based on the active keytype. The value passed to this function and entered into the handle attached to the input key item.

Arguments:	`CRSP_ITM_HNDL *hndl -`
	Pointer to the item handle containing the needed set structure information and the current item list.
	
	`Char *key_itm -`
	String containing an itm_name of an input key item to be loaded.
	
	`Void keyval -`
	Data to be loaded into the key item. Data must agree with the item's type.

Return Values:	`CRSP_SUCCESS:`
	If data loaded successfully
	
	`CRSP_FAIL:`
	If error in parameters, handle not open, key item.

Side Effects:	If successful, the keyval is copied into the data location for the input key item element in the handle.
Preconditions:	The item handle must be initialized and opened. The item key array must be initialized based on a keytype with the `crsp_itm_open` or `crsp_itm_init_key` functions. The key_itm must be a valid item for that keytype, and the keyval data must agree with the type of that item.

crsp_itm_get_key

Gets the key found by crsp_itm_read for a named key item.

Prototype:	`int crsp_itm_get_key (CRSP_ITM_HNDL *hndl, char *key_itm, void *keyval)`
Description:	Retrieves key information for data loaded by a `crsp_itm_read` call. An output key item list is prepared when the key is initialized, and loaded by `crsp_itm_read`. This function finds the key_itm in the list and copies ithe value into the user's location.
Arguments:	`CRSP_ITM_HNDL *hndl` – Pointer to the item handle containing the needed set structure information and the current item list. `Char *key_itm` – String containing an itm_name of a loaded key to be retrieved. `Void *keyval` – Location to place the key value. Location must agree with the item's type and size.
Return Values:	`CRSP_SUCCESS`: If data loaded successfully `CRSP_FAIL`: If error in parameters, handle not open, key item.
Side Effects:	If successful, the keyval is loaded based on the item and key value type.
Preconditions:	The item handle must be initialized and opened. The item key array must be initialized based on a keytype with the `crsp_irtm_open` or `crsp_itm_init_key` functions. The key_itm must be a valid item for that keytype, and the keyval data must agree with the type of that item.

crsp_itm_parse_key

Sets the key specifications to be used when selecting data based on a text string.

Prototype:	`int crsp_itm_parse_key (CRSP_ITM_HNDL *hndl, char *key_text)`
Description:	Loads key information in text format that will be used to load data in a `crsp_itm_read` call. The key is setup during `crsp_itm_open` based on the keytype identifier. The key_text string is parsed in the prescribed mapping order of the keytype and loaded into the handle. The key_text is in the form key1.key2…, where input key items are separated by periods. If an input key is not provided it will be set with a missing value. For example, if the keytype is gvkey, to access IBM company and security data of its primary security, key_text will be set to "6066.01". IID is optional if only company items are selected. In this case, "6066" may be used.
Arguments:	`CRSP_ITM_HNDL *hndl` – Pointer to the item handle containing the needed set structure information and the current item list. `Void *key_text` – String containing key information on interest, in order of keys, with each item separated by a period.
Return Values:	`CRSP_SUCCESS`: If successfully loaded. `CRSP_FAIL`: If bad parameter, handle not open, key item.
Side Effects:	If successful, the values are copied into the handle input key item data locations for each input key item from the key_text string.
Preconditions:	The item handle must be initialized and opened. The item key array must e initialized based on a keytype with the `crsp_itm_open` or `crsp_itm_init_key` functions.

crsp_itm_read

Loads data into a handle based on provided keys, supporting possible secondary indexes.

Prototype:	`int crsp_itm_read (CRSP_ITM_HNDL *hndl, int keyflag, init *status)`
Description:	Loads data from handle based on item keys specified in prior `crsp_itm_key` calls and keyflag. Depending on the level of the entity class, the operation may include reading data from the database into structures and/or specifying data already loaded. This allows a direct or positional read based on keyflag. If the handle `fiscal_disp_cd` is C, any fiscal-based time series are shifted to a calendar basis as part of the read operation.
Arguments:	`CRSP_ITM_HNDL *hndl` – Pointer to the item handle containing the needed set structure information and the current item list. `int keyflag` – Code determining how the key is interpreted. CRSP_EXACT to look for a specific value, CRSP_BACK ir CRSP_FORWARD for direct selection when partial matches are allowed, or a positional qualifier to base selection on the position relative to the last key accessed. Qualifiers include: • `CRSP_NEXT (=-99)` – read next key in sequence • `CRSP_PREV (=-96)` – read previous key in sequence • `CRSP_SAME (=-98)` – read same key, possibly with different information • `CRSP_FIRST (=-95)` – read first key in the database • `CRSP_LAST (=-97)` – read last key in the database `Int *status` – User provided location to load with the level of the read. It will be loaded with a 0 if the load results in reading new master data. It will be loaded with a number greater than 0 if the load impacts detail or global data, but no master data are affected.

Return Values:	`CRSP_SUCCESS:`
	If data loaded successfully
	
	`CRSP_EOF:`
	If positional read reaches the end of the file
	
	`CRSP_NOT_FOUND:`
	If key not found on exact read. If a detal input key is not provided and no items of that entity class are selected, the return is CRSP_SUCCESS as long as the primary key matches.
	
	`CRSP_FAIL:`
	If error in parameters, handle not opened, error in read operations.

Side Effects:	If successful, the wanted data for the key are loaded into the handle set structure which allows item objects to point to the loaded data. The key found for each level is loaded into the outkey item list. If the handle fiscal_disp_cd is set to calendar-based and items are fiscal-based, shifted calendars are created and time series are converted to calendar basis. The status argument is loaded based on whether the primary key changed. Handle primkey field and readlvl are set. Readlvl is set to the rank of the first entity class changed. If the primary key changed, getlvl is set to 0.
Preconditions:	The item handle must be initialized and opened. The item key must be initialized based on the key type, key element, and the entity class. If not a positional qualifier, the item key inpkey list must be loaded.

crsp_itm_close

Closes databases indicated by a handle, frees all item list structures, and frees the handle itself.

Prototype:	`int crsp_itm_close (CRSP_ITM_HNDL **hndl)`
Description:	Frees all item lists and item indexes, clears all calendar and key lists, closes the database, frees the handle set, and frees the handle itself. On completion, the handle will be set to NULL.
Arguments:	`CRSP_ITM_HNDL **hndl` – Pointer to the item handle to close.
Return Values:	`CRSP_SUCCESS`: If the database is successfully closed and all handle data are free `CRSP_FAIL`: If there is an error in the parameters, inconsistent handle, error closing databases.
Side Effects:	If successful, the handle data are emptied: • The itm_set database will be closed and the structure cleared. • The itm_grp, itm_idx, and itm_avail arrays will be emptied and all item lists will be freed. • Itm_key and cal_avail arrays will be freed. • The handle itself will be freed and its pointer set to NULL.
Preconditions:	The item handle must be previously opened with `crsp_itm_init`.

ITEM SELECTION FUNCTIONS

crsp_itm_load

Prepare a list from a full_list description string.

Prototype:	`int crsp_itm_load(CRSP_ITM_HNDL *hndl, char *full_list, int match_flag)`
Description:	Prepare items described by a full list and load them to an item table structure in an item handle. Splits the full list into the global section and the list section and uses `crsp_itm_expand_elem` on each list element in the list section. This well recursively expand the list elements to fill the structure and apply global qualifiers during the process.
Arguments:	`char *full_list` – Pointer to a string describing all items to add, used on standard item notation. `CRSP_ITM_HNDL *hndl` – Pointer to the item handle containing the needed set structure information and the current item list. `int match_flag` – Flag setting the behavior when an item if found but not the keyset. Values are: • `CRSP_MATCH_REQUIRED (=0)` – if any indicated item and keyset is not found no further items will be added, and CRSP_NOT_FOUND returned. • `CRSP_MATCH_FILL (=1)` – a dummy item will be created for any item if the item exists but the keyset dies not exist for that item in the current database. • `CRSP_MATCH_IGNORE (=2)` – items will not be added if the keyset is not found, but the return remains CRSP_SUCCESS.
Return Values:	`CRSP_SUCCESS`: If successful, and all indicated items loaded according to match_flag `CRSP_FAIL`: Error in parameters, bad list, handle not initialized, or reference data not available.
Side Effects:	If successful, the `CRSP_ITM_GRP` is loaded with all indicated items. A CRSP_ITM is allocated for each item/keyset pair not already loaded. Object pointers are not set by this function.
Preconditions:	The item handle set must be loaded. The item table must be initialized with an available app_id. The first set in the set list must agree with the app_id.

crsp_itm_load_file

Prepare a list from an item list description file

Prototype:	`int crsp_itm_load_file (CRSP_ITM_HNDL *hndl, char *file_path, char *gbl_list, int match_flag)`
Description:	Prepare items described by a listfile and load them to an item table structure in an item handle. Identifies a global section and uses `crsp_itm_load_elem` on each list element in the file. This will recursively expand the list elements to fill the structure and apply global qualifiers during the process.
Arguments:	`char *file_path` – pointer to a string containing an input file of data items. Each row in the input file must be a list element. `char *gbl_list` – pointer to a string containing the global information to be applied to all list elements. `CRSP_ITM_HNDL *hndl` – pointer to the item handle containing the needed set structure information and the current item list. `int match_flag` – Flag setting the behavior when an item is found but not the keyset. Values are: • `CRSP_MATCH_REQUIRED (=0)` – if any indicated item and keyset is not found no further items will be added, and CRSP_NOT_FOUND returned. • `CRSP_MATCH_FILL (=1)` – a dummy item will be created for any item if the item exists but the keyset dies not exist for that item in the current database. • `CRSP_MATCH_IGNORE (=2)` – items will not be added if the keyset is not found, but the return remains CRSP_SUCCESS.

Return Values:	`CRSP_SUCCESS:`
	If successful
	`CRSP_FAIL:`
	If error in parameters, bad list, handle not initialized, or reference data not available.

Side Effects:	If successful, the `CRSP_ITM_GRP` is loaded with all indicated items. A `CRSP_ITM` is allocated for each item/keyset pair not already loaded. Object pointers are not set by this function.
Preconditions:	The item handle set must be loaded. The item table must be initialized with an available app_id. The first set in the set list must agree with the app_id. The input file must exist with one list element per row.

crsp_itm_load_printopt

Prepare a list from a groupname print option code.

Prototype:	`int crsp_itm_load_printopt (CRSP_ITM_HNDL *hndl, char *printopt, int match_flag)`
Description:	Prepare items described by a print option code describing one group and load them to an item table structure in an item handle.
Arguments:	`char *printopt` – pointer to a string containing a print option code in the form xx[.keyset_string]. `CRSP_ITM_HNDL *hndl` – pointer to the item handle containing the needed set structure information and the current item list. `int match_flag` – flag setting the behavior when an item if found but not in the keyset Values are: • `CRSP_MATCH_REQUIRED (=0)` – if any indicated item and keyset is not found no further items will be added, and CRSP_NOT_FOUND returned. • `CRSP_MATCH_FILL (=1)` – a dummy item will be created for any item if the item exists but the keyset dies not exist for that item in the current database. • `CRSP_MATCH_IGNORE (=2)` – items will not be added if the keyset is not found, but the return remains CRSP_SUCCESS.

Return Values:	CRSP_SUCCESS: if successful. CRSP_FAIL: Error in parameters, opening input file, bad format of global list or input file, or reference data not available.
Side Effects:	If successful, the CRSP_ITM_GRP is loaded with all indicated items. A CRSP_ITM is allocated for each item/keyset pair not already loaded. Object pointers are not set by this function.
Preconditions:	The item handle set must be loaded. The item table must be initialized with an available app_id. The first set in the set list must agree with the app_id. The 2-letter print option code must be known to the app_id. Only groups with grptype of S or D will have non-blank printopt codes available.

crsp_itm_find

Access an individual item that was loaded.

Prototype:	int crsp_itm_find (CRSP_ITM_HNDL *hndl, char *itm_name, int keyset, CRSP_ITM **foundptr)
Description:	Attach a pointer to a CRSP_ITM that was previously loaded. The CRSP_ITM structure describes the data item and contains the underlying time series, array, or row data.
Arguments:	CRSP_ITM_HNDL *hndl – Pointer to the item handle containing the needed set structure information and the current item list. char *itm_name – String containing the itm_name to find. int keyset – Keyset to find CRSP_ITM **foundptr – Pointer that will point to the item data found.
Return Values:	CRSP_SUCCESS: If successful CRSP NOT FOUND: If the itm_name and keyset combination are not available CRSP_FAIL: If error in parameters, handle not initialized, or error searching for the item.
Side Effects:	If successful, the foundptr will point to a CRSP_ITM with data and information for the desired item and keyset.
Preconditions:	The item handle set must be initialized, loaded with a list of items, and opened.

crsp_itm_free_list

Resets all item lists previously loaded into a handle.

Prototype:	int crsp_itm_free_list (CRSP_ITM_HNDL *hndl)
Description:	Resets the handle by freeing all item lists and item indexes.
Arguments:	CRSP_ITM_HNDL *hndl – pointer to the item handle to reset.
Return Values:	CRSP_SUCCESS: If successfully frees the data CRSP_FAIL: If error in parameters, inconsistent handle, error emptying the lists.
Side Effects:	If successful, the item lists are emptied: itm_list, keyset_list, struct_list. The index arrays are also emptied. New items can be loaded.
Preconditions:	The item handle must be previously opened with crsp_itm_init.

crsp_itm_is_miss_arrval

Check if a value from a data-object attached to the item is a missing value

Prototype:	`int crsp_itm_is_miss_arrval (CRSP_ITM *itm, int ind*is_miss)`
Description:	Checks if the requested element in a data-object attached to the item contains a missing value. is_miss is set to 1 when missing value is detected. Only items of simple (non-structured) types are accepted, while the item's underlying data-object ca be of structured data-type, in which case the structure offset is used to extract the item value.
Arguments:	`CRSP_ITM *itm` – Pointer to the item `int ind` – Index of the data array element to check `int *is_miss` – Pointer to the resulting flag value
Return Values:	CRSP_SUCCESS: If successful, the returned value is initialized and set. CRSP_FAIL: If error in parameters, bad item or element index is out-of-range (ignored in case of CRSP_ROW)
Side Effects:	
Preconditions:	The item has to have a valid bound data-object. Structured items are not allowed. Field items of structures are allowed.