Access to Astronomical Catalogues

← Click to display the menu


```
I/259          The Tycho-2 Catalogue              (Hog+ 2000)
```

```
The Tycho-2 Catalogue of the 2.5 Million Brightest Stars
    Hog E., Fabricius C., Makarov V.V., Urban S., Corbin T.,
    Wycoff G., Bastian U., Schwekendiek P., Wicenec A.
   <Astron. Astrophys. 355, L27 (2000)>
   =2000A&A...355L..27H
```

```
ADC_Keywords: Positional data ; Proper motions ; Surveys ;
              Photometry ; Stars, double and multiple
Mission_Name: Hipparcos

Keywords: astrometry - stars: fundamental parameters - catalogs


Abstract:
    The Tycho-2 Catalogue is an astrometric reference catalogue containing
    positions  and proper motions  as well as  two-colour photometric data
    for the 2.5 million brightest stars in the sky.  The Tycho-2 positions
    and magnitudes are  based on  precisely the  same observations  as the
    original  Tycho  Catalogue  (hereafter  Tycho-1;  see  Cat.  I/239))
    collected  by the  star mapper  of the  ESA Hipparcos  satellite,  but
    Tycho-2 is much bigger  and slightly  more precise,   owing to  a more
    advanced  reduction technique.  Components  of  double  stars  with
    separations down to 0.8 arcsec are included. Proper motions precise to
    about 2.5 mas/yr are given as  derived  from  a  comparison  with  the
    Astrographic  Catalogue  and  143  other  ground-based  astrometric
    catalogues,  all reduced to the Hipparcos celestial coordinate system.
    Tycho-2  supersedes in most applications Tycho-1,   as well as the ACT
    (Cat.  I/246)  and TRC  (Cat. I/250)  catalogues based on Tycho-1.
    Supplement-1  lists stars  from the  Hipparcos and  Tycho-1 Catalogues
    which are not in Tycho-2.  Supplement-2 lists 1146 Tycho-1 stars which
    are probably either false or heavily disturbed.

    For more information, please consult the Tycho-2 home page:
    http://www.astro.ku.dk/~erik/Tycho-2

Catalogue Characteristics:

    The principal characteristics of the Tycho-2 Catalogue are
    summarized below. By means of proper motions the positions
    are transferred to the year 2000.0, the epoch of the catalogue.
    The median values of internal standard errors are given.
    ------------------------------------------------------------
    Mean satellite observation epoch      ~J1991.5
    Epoch of the Tycho-2 Catalogue         J2000.0
    Reference system                       ICRS
      coincidence with ICRS (1)             ±0.6 mas
      deviation from inertial (1)           ±0.25 mas/yr
    Number of entries                      2,539,913
    Astrometric standard errors (2)
      V_T < 9 mag                          7 mas
      all stars, positions                   60 mas
      all stars, proper motions              2.5 mas/yr
    Photometric std. errors (3) on V_T
      V_T < 9 mag                          0.013 mag
      all stars                              0.10 mag
    Star density
      b= 0 deg                             150 stars/sq.deg.
      b= ±30 deg                          50 stars/sq.deg.
      b= ±90 deg                          25 stars/sq.deg.
    Completeness to 90 per cent            V ~ 11.5 mag
    Completeness to 99 per cent            V ~ 11.0 mag
    Number of Tycho observations           ~300 10^6
    ------------------------------------------------------------
    Note (1): about all 3 axes
    Note (2):
        ratio of external to internal standard errors is ~1.0
        for positions and for proper motions. Systematic errors
        are less than 1 mas and 0.5 mas/yr
    Note (3):
        ratio of photometric external to internal standard errors
        at V_T > 9 mag is below 1.5
```

File Summary:

FileName	Lrecl	Records	Explanations
ReadMe	80	.	This file
tyc2.dat	206	2539913	*The Tycho-2 main catalogue
suppl_1.dat	122	17588	The Tycho-2 supplement-1
suppl_2.dat	122	1146	The Tycho-2 supplement-2
index.dat	42	9538	Index to Tycho-2 and supplement-1
guide.ps	120	7837	Guide to Tycho-2 (postscript)
guide.pdf	1	428724	Guide to Tycho-2 (pdf)

Note on tyc2.dat:
This huge file is split into 20 smaller files named tyc2.dat.00, tyc2.dat.01, ... tyc2.dat.18, each containing 127000 lines, and tyc2.dat.19 which contains the last 126913 lines.

See also:
- I/239 : The Hipparcos and Tycho Catalogues (ESA 1997)
- I/211 : CCDM (Components of Double and Multiple stars) (Dommanget+ 1994)
- I/246 : The ACT Reference Catalog (Urban+ 1997)
- I/250 : The Tycho Reference Catalogue (Hog+ 1998)
- http://www.astro.ku.dk/~erik/Tycho-2 : Tycho-2 home page

Nomenclature Notes:
The TYC identifier is constructed from the GSC region number (TYC1), the running number within the region (TYC2) and a component identifier (TYC3) which is normally 1. Some non-GSC running numbers were constructed for the first Tycho Catalogue and for Tycho-2.
The recommended star designation contains a hyphen between the TYC numbers, e.g. TYC 1-13-1.

Field separator in the data files:
In the data files, the fields in each record are separated by a vertical bar, |. In this connection the TYC identifier (TYC1, TYC2 and TYC3) constitutes one field and the pair of a HIP number with a CCDM identifier is also considered one field.

Byte-by-byte Description of file: tyc2.dat

Bytes	Format	Units	Label	Explanations
1- 4	I4	---	TYC1	[1,9537]+= TYC1 from TYC or GSC (1)
6- 10	I5	---	TYC2	[1,12121] TYC2 from TYC or GSC (1)
12	I1	---	TYC3	[1,3] TYC3 from TYC (1)
14	A1	---	pflag	[PX] mean position flag (2)
16- 27	F12.8	deg	RAmdeg	[]? Mean Right Asc, ICRS, epoch=J2000 (3)
29- 40	F12.8	deg	DEmdeg	[]? Mean Decl, ICRS, at epoch=J2000 (3)
42- 48	F7.1	mas/yr	pmRA	? Proper motion in RA*cos(dec) (12)
50- 56	F7.1	mas/yr	pmDE	? Proper motion in Dec (12)
58- 60	I3	mas	e_RAmdeg	[3,183]? s.e. RA*cos(dec),at mean epoch (5)
62- 64	I3	mas	e_DEmdeg	[1,184]? s.e. of Dec at mean epoch (5)
66- 69	F4.1	mas/yr	e_pmRA	[0.2,11.5]? s.e. prop mot in RA*cos(dec)(5)
71- 74	F4.1	mas/yr	e_pmDE	[0.2,10.3]? s.e. of proper motion in Dec(5)
76- 82	F7.2	yr	EpRAm	[1915.95,1992.53]? mean epoch of RA (4)
84- 90	F7.2	yr	EpDEm	[1911.94,1992.01]? mean epoch of Dec (4)
92- 93	I2	---	Num	[2,36]? Number of positions used
95- 97	F3.1	---	q_RAmdeg	[0.0,9.9]? Goodness of fit for mean RA (6)
99-101	F3.1	---	q_DEmdeg	[0.0,9.9]? Goodness of fit for mean Dec (6)
103-105	F3.1	---	q_pmRA	[0.0,9.9]? Goodness of fit for pmRA (6)
107-109	F3.1	---	q_pmDE	[0.0,9.9]? Goodness of fit for pmDE (6)
111-116	F6.3	mag	BTmag	[2.183,16.581]? Tycho-2 BT magnitude (7)
118-122	F5.3	mag	e_BTmag	[0.014,1.977]? s.e. of BT (7)
124-129	F6.3	mag	VTmag	[1.905,15.193]? Tycho-2 VT magnitude (7)
131-135	F5.3	mag	e_VTmag	[0.009,1.468]? s.e. of VT (7)
137-139	I3	0.1arcsec	prox	[3,999] proximity indicator (8)
141	A1	---	TYC	[T] Tycho-1 star (9)
143-148	I6	---	HIP	[1,120404]? Hipparcos number
149-151	A3	---	CCDM	CCDM component identifier for HIP stars(10)
153-164	F12.8	deg	RAdeg	Observed Tycho-2 Right Ascension, ICRS
166-177	F12.8	deg	DEdeg	Observed Tycho-2 Declination, ICRS
179-182	F4.2	yr	EpRA-1990	[0.81,2.13] epoch-1990 of RAdeg
184-187	F4.2	yr	EpDE-1990	[0.72,2.36] epoch-1990 of DEdeg
189-193	F5.1	mas	e_RAdeg	s.e.RA*cos(dec), of observed Tycho-2 RA (5)

```
 195-199  F5.1    mas     e_DEdeg  s.e. of observed Tycho-2 Dec (5)
     201  A1      ---     posflg   [ DP] type of Tycho-2 solution (11)
 203-206  F4.1    ---     corr     [-1,1] correlation (RAdeg,DEdeg)
```

```
Note (1): The TYC identifier is constructed from the GSC region number
    (TYC1), the running number within the region (TYC2) and a component
    identifier (TYC3) which is normally 1. Some non-GSC running numbers
    were constructed for the first Tycho Catalogue and for Tycho-2.
    The recommended star designation contains a hyphen between the
    TYC numbers, e.g. TYC 1-13-1.
Note (2):
    ' ' = normal mean position and proper motion.
    'P' = the mean position, proper motion, etc., refer to the
          photocentre of two Tycho-2 entries, where the BT magnitudes
          were used in weighting the positions.
    'X' = no mean position, no proper motion.
Note (3):
    The mean position is a weighted mean for the catalogues contributing
    to the proper motion determination. This mean has then been brought to
    epoch 2000.0 by the computed proper motion. See Note(2) above for
    details. Tycho-2 is one of the several catalogues used to determine
    the mean position and proper motion. The observed Tycho-2 position is
    given in the fields RAdeg and DEdeg.
Note (4):
    The mean epochs are given in Julian years.
Note (5):
    The errors are based on error models.
Note (6):
    This goodness of fit is the ratio of the scatter-based and the
    model-based error. It is only defined when Num > 2. Values
    exceeding 9.9 are truncated to 9.9.
Note (7):
    Blank when no magnitude is available. Either BTmag or VTmag is
    always given. Approximate Johnson photometry may be obtained as:
    V   = VT -0.090*(BT-VT)
    B-V = 0.850*(BT-VT)
    Consult Sect 1.3 of Vol 1 of "The Hipparcos and Tycho Catalogues",
    ESA SP-1200, 1997, for details.
Note (8):
    Distance in units of 100 mas to the nearest entry in the Tycho-2
    main catalogue or supplement. The distance is computed for the
    epoch 1991.25. A value of 999 (i.e. 99.9 arcsec) is given if the
    distance exceeds 99.9 arcsec.
Note (9):
    ' ' = no Tycho-1 star was found within 0.8 arcsec (quality 1-8)
          or 2.4 arcsec (quality 9).
    'T' = this is a Tycho-1 star. The Tycho-1 identifier is given in the
          beginning of the record. For Tycho-1 stars, resolved in
          Tycho-2 as a close pair, both components are flagged as
          a Tycho-1 star and the Tycho-1 TYC3 is assigned to the
          brightest (VT) component.
    The HIP-only stars given in Tycho-1 are not flagged as Tycho-1 stars.
Note (10):
    The CCDM component identifiers for double or multiple Hipparcos stars
    contributing to this Tycho-2 entry. For photocentre solutions, all
    components within 0.8 arcsec contribute. For double star solutions any
    unresolved component within 0.8 arcsec contributes. For single star
    solutions, the predicted signal from close stars were normally
    subtracted in the analysis of the photon counts and such stars
    therefore do not contribute to the solution. The components are given
    in lexical order.
Note (11):
    ' ' = normal treatment, close stars were subtracted when possible.
    'D' = double star treatment. Two stars were found. The companion is
          normally included as a separate Tycho-2 entry, but may have
          been rejected.
    'P' = photocentre treatment, close stars were not subtracted. This
          special treatment was applied to known or suspected doubles
          which were not successfully (or reliably) resolved in the
          Tycho-2 double star processing.
Note (12): Some Hipparcos stars (having a positive number in the HIP column)
     have no proper motions; these are virtually all in multiple systems.
```

Byte-by-byte Description of file: suppl_1.dat, suppl_2.dat

```
   Bytes Format   Units   Label    Explanations
```

```
   1-  4  I4      ---     TYC1     [2,9529]+= TYC1 from TYC (1)
   6- 10  I5      ---     TYC2     [1,12112]  TYC2 from TYC (1)
```

```
      12  I1      ---     TYC3     [1,4]      TYC3 from TYC (1)
      14  A1      ---     flag     [HT] data from Hipparcos or Tycho-1 (2)
  16- 27  F12.8   deg     RAdeg    Right Asc, ICRS, at epoch=J1991.25
  29- 40  F12.8   deg     DEdeg    Decl, ICRS, at epoch=J1991.25
  42- 48  F7.1    mas/yr  pmRA     []? Proper motion in RA*cos(dec)
  50- 56  F7.1    mas/yr  pmDE     []? Proper motion in Dec
  58- 62  F5.1    mas     e_RAdeg  s.e. RA*cos(dec)
  64- 68  F5.1    mas     e_DEdeg  s.e. of Dec
  70- 74  F5.1    mas/yr  e_pmRA   []? s.e. prop mot in RA * cos(dec)
  76- 80  F5.1    mas/yr  e_pmDE   []? s.e. of proper motion in Dec
      82  A1      ---     mflag    [ BVH] Note about Tycho magnitudes (3)
  84- 89  F6.3    mag     BTmag    []? Tycho-1 BT magnitude (4)
  91- 95  F5.3    mag     e_BTmag  []? s.e. of BT (4)
  97-102  F6.3    mag     VTmag    []?  Tycho-1 VT or Hp magnitude (4)
 104-108  F5.3    mag     e_VTmag  []? s.e. of VT (4)
 110-112  I3    0.1arcsec prox     [1,999] proximity indicator (5)
     114  A1      ---     TYC      [ T] Tycho-1 star
 116-121  I6      ---     HIP      [1,120404]? Hipparcos number
     122  A1      ---     CCDM     CCDM component identifier for HIP stars
```

Note (1): The TYC identifier is constructed from the GSC region number (TYC1),
the running number within the region (TYC2) and a component identifier
(TYC3) which is normally 1. The numbers are copied from Tycho-1.
(see the "Nomenclature Notes" section above)

Note (2):
'H' = data are from Hipparcos and include proper motion.
'T' = data are from Tycho-1. No proper motion is given.

Note (3):
' ' = both BT and VT given.
'B' = only BT given.
'V' = only VT given.
'H' = Hp is given instead of VT. BT is blank.

Note (4):
Blank when no magnitude is available.
For Hipparcos stars with no VT, Hp is given instead of VT, and BT is blank.

Note (5):
Distance in units of 100 mas to nearest Tycho-2 main or supplement
entry. (Computed for the epoch 1991.25). A value of 999 (i.e. 99.9
arcsec) is given if the distance exceeds 99.9 arcsec.

Byte-by-byte Description of file: index.dat

```
   Bytes Format  Units   Label     Explanations

   1-  7  I7     ---     rec_t2    +  Tycho-2 rec. of 1st star in region (1)
   9- 14  I6     ---     rec_s1    += Suppl-1 rec. of 1st star in region (1)
  16- 21  F6.2   deg     RAmin     [-0.01,] smallest RA in region (2)
  23- 28  F6.2   deg     RAmax     [,360.00] largest RA in region (2)
  30- 35  F6.2   deg     DEmin     smallest Dec in this region (2)
  37- 42  F6.2   deg     DEmax     largest Dec in this region (2)
```

Note (1): The catalogue is sorted according to the GSC region numbers.
The line i of the index file gives the record number in Tycho-2 of
the first star in GSC region i. Line i+1 gives the record number +1
of the last star in GSC region i. For Supplement-1, some regions are
empty and line i and line i+1 give the same record number.

Note (2): a safe rounding was applied. Minimum values are always
rounded down and maximum values up.

References:

Hog E., Fabricius C., Makarov V.V., Bastian U., Schwekendiek P., Wicenec A., Urban S., Corbin T., Wycoff G., "Construction and verification of the Tycho-2 Catalogue." =2000A&A...357..367H

(End) C. Fabricius [Niels Bohr Institute] 19-Jan-2000

*The document above follows the rules of the Standard Description for Astronomical Catalogues.From this documentation it is possible to generate **f77** program to load files into arrays or line by line*

FORTRAN Generation (/./ftp/cats/I/259)

*Conversion of standardized `ReadMe` file for file **/./ftp/cats/I/259** into FORTRAN code for loading all data files into arrays.*

Note that special values are assigned to unknown or unspecified numbers (also called NULL numbers); when necessary, the coordinate components making up the right ascension and declination are converted into floating-point numbers representing these angles in degrees.

```
      program load_ReadMe
C=============================================================================
C  F77-compliant program generated by readme2f_1.81 (2015-09-23), on 2016-Jun-09
C=============================================================================
*  This code was generated from the ReadMe file documenting a catalogue
*  according to the "Standard for Documentation of Astronomical Catalogues"
*  currently in use by the Astronomical Data Centers (CDS, ADC, A&A)
*  (see full documentation at URL http://vizier.u-strasbg.fr/doc/catstd.htx)
*  Please report problems or questions to   question(at)simbad.u-strasbg.fr
C=============================================================================

      implicit none
*  Unspecified or NULL values, generally corresponding to blank columns,
*  are assigned one of the following special values:
*     rNULL__    for unknown or NULL floating-point values
*     iNULL__    for unknown or NULL   integer      values
      real*4     rNULL__
      integer*4  iNULL__
      parameter  (rNULL__=--2147483648.)        ! NULL real number
      parameter  (iNULL__=(-2147483647-1))      ! NULL int  number
      integer    idig                   ! testing NULL number

C=============================================================================
Cat. I/259           The Tycho-2 Catalogue              (Hog+ 2000)
*================================================================================
*The Tycho-2 Catalogue of the 2.5 Million Brightest Stars
*    Hog E., Fabricius C., Makarov V.V., Urban S., Corbin T.,
*    Wycoff G., Bastian U., Schwekendiek P., Wicenec A.
*   <Astron. Astrophys. 355, L27 (2000)>
*   =2000A&A...355L..27H
C=============================================================================

C  Internal variables

      integer*4 i__

c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

C  Declarations for 'tyc2.dat'  ! *The Tycho-2 main catalogue

      integer*4 nr__
      parameter (nr__=2539913)  ! Number of records
      character*206 ar__        ! Full-size record

C  J2000 position composed of: RAdeg DEdeg
      integer*4     TYC1       (nr__) ! [1,9537]+= TYC1 from TYC or GSC (1)
      integer*4     TYC2       (nr__) ! [1,12121]  TYC2 from TYC or GSC (1)
      integer*4     TYC3       (nr__) ! [1,3]      TYC3 from TYC (1)
      character*1   pflag      (nr__) ! [ PX] mean position flag (2)
      real*8        RAmdeg     (nr__) ! (deg) []? Mean Right Asc, ICRS, epoch=J2000 (3)
      real*8        DEmdeg     (nr__) ! (deg) []? Mean Decl, ICRS, at epoch=J2000 (3)
      real*8        pmRA       (nr__) ! (mas/yr) ? Proper motion in RA*cos(dec) (12)
      real*8        pmDE       (nr__) ! (mas/yr) ? Proper motion in Dec (12)
      integer*4     e_RAmdeg   (nr__) ! (mas) [3,183]? s.e. RA*cos(dec),at mean epoch (5)
      integer*4     e_DEmdeg   (nr__) ! (mas) [1,184]? s.e. of Dec at mean epoch (5)
      real*4        e_pmRA     (nr__) ! (mas/yr) [0.2,11.5]? s.e. prop mot in RA*cos(dec)(5)
      real*4        e_pmDE     (nr__) ! (mas/yr) [0.2,10.3]? s.e. of proper motion in Dec(5)
      real*8        EpRAm      (nr__) ! (yr) [1915.95,1992.53]? mean epoch of RA (4)
      real*8        EpDEm      (nr__) ! (yr) [1911.94,1992.01]? mean epoch of Dec (4)
      integer*4     Num        (nr__) ! [2,36]? Number of positions used
      real*4        q_RAmdeg   (nr__) ! [0.0,9.9]? Goodness of fit for mean RA (6)
      real*4        q_DEmdeg   (nr__) ! [0.0,9.9]? Goodness of fit for mean Dec (6)
      real*4        q_pmRA     (nr__) ! [0.0,9.9]? Goodness of fit for pmRA (6)
      real*4        q_pmDE     (nr__) ! [0.0,9.9]? Goodness of fit for pmDE (6)
      real*4        BTmag      (nr__) ! (mag) [2.183,16.581]? Tycho-2 BT magnitude (7)
      real*4        e_BTmag    (nr__) ! (mag) [0.014,1.977]? s.e. of BT (7)
      real*4        VTmag      (nr__) ! (mag) [1.905,15.193]? Tycho-2 VT magnitude (7)
      real*4        e_VTmag    (nr__) ! (mag) [0.009,1.468]? s.e. of VT (7)
      integer*4     prox       (nr__) ! (0.1arcsec) [3,999] proximity indicator (8)
```

```
      character*1   TYC        (nr__) ! [T] Tycho-1 star (9)
      integer*4     HIP        (nr__) ! [1,120404]? Hipparcos number
      character*3   CCDM       (nr__) ! CCDM component identifier for HIP stars(10)
      real*8        RAdeg      (nr__) ! (deg) Observed Tycho-2 Right Ascension, ICRS
      real*8        DEdeg      (nr__) ! (deg) Observed Tycho-2 Declination, ICRS
      real*4        EpRA_1990  (nr__) ! (yr) [0.81,2.13]  epoch-1990 of RAdeg
      real*4        EpDE_1990  (nr__) ! (yr) [0.72,2.36]  epoch-1990 of DEdeg
      real*4        e_RAdeg    (nr__) ! (mas) s.e.RA*cos(dec), of observed Tycho-2 RA (5)
      real*4        e_DEdeg    (nr__) ! (mas) s.e. of observed Tycho-2 Dec (5)
      character*1   posflg     (nr__) ! [ DP] type of Tycho-2 solution (11)
      real*4        corr       (nr__) ! [-1,1] correlation (RAdeg,DEdeg)
*Note (1): The TYC identifier is constructed from the GSC region number
*    (TYC1), the running number within the region (TYC2) and a component
*    identifier (TYC3) which is normally 1. Some non-GSC running numbers
*    were constructed for the first Tycho Catalogue and for Tycho-2.
*    The recommended star designation contains a hyphen between the
*    TYC numbers, e.g. TYC 1-13-1.
*Note (2):
*    ' ' = normal mean position and proper motion.
*    'P' = the mean position, proper motion, etc., refer to the
*          photocentre of two Tycho-2 entries, where the BT magnitudes
*          were used in weighting the positions.
*    'X' = no mean position, no proper motion.
*Note (3):
*    The mean position is a weighted mean for the catalogues contributing
*    to the proper motion determination. This mean has then been brought to
*    epoch 2000.0 by the computed proper motion. See Note(2) above for
*    details. Tycho-2 is one of the several catalogues used to determine
*    the mean position and proper motion. The observed Tycho-2 position is
*    given in the fields RAdeg and DEdeg.
*Note (4):
*    The mean epochs are given in Julian years.
*Note (5):
*    The errors are based on error models.
*Note (6):
*    This goodness of fit is the ratio of the scatter-based and the
*    model-based error. It is only defined when Num > 2. Values
*    exceeding 9.9 are truncated to 9.9.
*Note (7):
*    Blank when no magnitude is available. Either BTmag or VTmag is
*    always given. Approximate Johnson photometry may be obtained as:
*    V   = VT -0.090*(BT-VT)
*    B-V = 0.850*(BT-VT)
*    Consult Sect 1.3 of Vol 1 of "The Hipparcos and Tycho Catalogues",
*    ESA SP-1200, 1997, for details.
*Note (8):
*    Distance in units of 100 mas to the nearest entry in the Tycho-2
*    main catalogue or supplement. The distance is computed for the
*    epoch 1991.25. A value of 999 (i.e. 99.9 arcsec) is given if the
*    distance exceeds 99.9 arcsec.
*Note (9):
*    ' ' = no Tycho-1 star was found within 0.8 arcsec (quality 1-8)
*          or 2.4 arcsec (quality 9).
*    'T' = this is a Tycho-1 star. The Tycho-1 identifier is given in the
*          beginning of the record. For Tycho-1 stars, resolved in
*          Tycho-2 as a close pair, both components are flagged as
*          a Tycho-1 star and the Tycho-1 TYC3 is assigned to the
*          brightest (VT) component.
*    The HIP-only stars given in Tycho-1 are not flagged as Tycho-1 stars.
*Note (10):
*    The CCDM component identifiers for double or multiple Hipparcos stars
*    contributing to this Tycho-2 entry. For photocentre solutions, all
*    components within 0.8 arcsec contribute. For double star solutions any
*    unresolved component within 0.8 arcsec contributes. For single star
*    solutions, the predicted signal from close stars were normally
*    subtracted in the analysis of the photon counts and such stars
*    therefore do not contribute to the solution. The components are given
*    in lexical order.
*Note (11):
*    ' ' = normal treatment, close stars were subtracted when possible.
*    'D' = double star treatment. Two stars were found. The companion is
*          normally included as a separate Tycho-2 entry, but may have
*          been rejected.
*    'P' = photocentre treatment, close stars were not subtracted. This
*          special treatment was applied to known or suspected doubles
*          which were not successfully (or reliably) resolved in the
*          Tycho-2 double star processing.
*Note (12): Some Hipparcos stars (having a positive number in the HIP column)
*     have no proper motions; these are virtually all in multiple systems.

c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

C  Declarations for 'suppl_1.dat'       ! The Tycho-2 supplement-1
```

```
      integer*4 nr__1
      parameter (nr__1=17588)   ! Number of records
      character*122 ar__1       ! Full-size record

C  J2000 position composed of: RAdeg DEdeg (Epoch=J1991.25)
      integer*4     TYC1_1     (nr__1) ! [2,9529]+= TYC1 from TYC (1)
      integer*4     TYC2_1     (nr__1) ! [1,12112]  TYC2 from TYC (1)
      integer*4     TYC3_1     (nr__1) ! [1,4]      TYC3 from TYC (1)
      character*1   flag       (nr__1) ! [HT] data from Hipparcos or Tycho-1 (2)
      real*8        RAdeg_1    (nr__1) ! (deg) Right Asc, ICRS, at epoch=J1991.25
      real*8        DEdeg_1    (nr__1) ! (deg) Decl, ICRS, at epoch=J1991.25
      real*8        pmRA_1     (nr__1) ! (mas/yr) []? Proper motion in RA*cos(dec)
      real*8        pmDE_1     (nr__1) ! (mas/yr) []? Proper motion in Dec
      real*4        e_RAdeg_1  (nr__1) ! (mas) s.e. RA*cos(dec)
      real*4        e_DEdeg_1  (nr__1) ! (mas) s.e. of Dec
      real*4        e_pmRA_1   (nr__1) ! (mas/yr) []? s.e. prop mot in RA * cos(dec)
      real*4        e_pmDE_1   (nr__1) ! (mas/yr) []? s.e. of proper motion in Dec
      character*1   mflag      (nr__1) ! [ BVH] Note about Tycho magnitudes (3)
      real*4        BTmag_1    (nr__1) ! (mag) []? Tycho-1 BT magnitude (4)
      real*4        e_BTmag_1  (nr__1) ! (mag) []? s.e. of BT (4)
      real*4        VTmag_1    (nr__1) ! (mag) []?  Tycho-1 VT or Hp magnitude (4)
      real*4        e_VTmag_1  (nr__1) ! (mag) []? s.e. of VT (4)
      integer*4     prox_1     (nr__1) ! (0.1arcsec) [1,999] proximity indicator (5)
      character*1   TYC_1      (nr__1) ! [ T] Tycho-1 star
      integer*4     HIP_1      (nr__1) ! [1,120404]? Hipparcos number
      character*1   CCDM_1     (nr__1) ! CCDM component identifier for HIP stars
*Note (1): The TYC identifier is constructed from the GSC region number (TYC1),
*    the running number within the region (TYC2) and a component identifier
*    (TYC3) which is normally 1. The numbers are copied from Tycho-1.
*    (see the "Nomenclature Notes" section above)
*Note (2):
*    'H' = data are from Hipparcos and include proper motion.
*    'T' = data are from Tycho-1. No proper motion is given.
*Note (3):
*    ' ' = both BT and VT given.
*    'B' = only BT given.
*    'V' = only VT given.
*    'H' = Hp is given instead of VT. BT is blank.
*Note (4):
*    Blank when no magnitude is available.
*    For Hipparcos stars with no VT, Hp is given instead of VT, and BT is blank.
*Note (5):
*    Distance in units of 100 mas to nearest Tycho-2 main or supplement
*    entry. (Computed for the epoch 1991.25). A value of 999 (i.e. 99.9
*    arcsec) is given if the distance exceeds 99.9 arcsec.

c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

C  Declarations for 'suppl_2.dat'       ! The Tycho-2 supplement-2

      integer*4 nr__2
      parameter (nr__2=1146)    ! Number of records
      character*122 ar__2       ! Full-size record

C  J2000 position composed of: RAdeg DEdeg (Epoch=J1991.25)
      integer*4     TYC1_2     (nr__2) ! [2,9529]+= TYC1 from TYC (1)
      integer*4     TYC2_2     (nr__2) ! [1,12112]  TYC2 from TYC (1)
      integer*4     TYC3_2     (nr__2) ! [1,4]      TYC3 from TYC (1)
      character*1   flag_1     (nr__2) ! [HT] data from Hipparcos or Tycho-1 (2)
      real*8        RAdeg_2    (nr__2) ! (deg) Right Asc, ICRS, at epoch=J1991.25
      real*8        DEdeg_2    (nr__2) ! (deg) Decl, ICRS, at epoch=J1991.25
      real*8        pmRA_2     (nr__2) ! (mas/yr) []? Proper motion in RA*cos(dec)
      real*8        pmDE_2     (nr__2) ! (mas/yr) []? Proper motion in Dec
      real*4        e_RAdeg_2  (nr__2) ! (mas) s.e. RA*cos(dec)
      real*4        e_DEdeg_2  (nr__2) ! (mas) s.e. of Dec
      real*4        e_pmRA_2   (nr__2) ! (mas/yr) []? s.e. prop mot in RA * cos(dec)
      real*4        e_pmDE_2   (nr__2) ! (mas/yr) []? s.e. of proper motion in Dec
      character*1   mflag_1    (nr__2) ! [ BVH] Note about Tycho magnitudes (3)
      real*4        BTmag_2    (nr__2) ! (mag) []? Tycho-1 BT magnitude (4)
      real*4        e_BTmag_2  (nr__2) ! (mag) []? s.e. of BT (4)
      real*4        VTmag_2    (nr__2) ! (mag) []?  Tycho-1 VT or Hp magnitude (4)
      real*4        e_VTmag_2  (nr__2) ! (mag) []? s.e. of VT (4)
      integer*4     prox_2     (nr__2) ! (0.1arcsec) [1,999] proximity indicator (5)
      character*1   TYC_2      (nr__2) ! [ T] Tycho-1 star
      integer*4     HIP_2      (nr__2) ! [1,120404]? Hipparcos number
      character*1   CCDM_2     (nr__2) ! CCDM component identifier for HIP stars
*Note (1): The TYC identifier is constructed from the GSC region number (TYC1),
*    the running number within the region (TYC2) and a component identifier
*    (TYC3) which is normally 1. The numbers are copied from Tycho-1.
*    (see the "Nomenclature Notes" section above)
*Note (2):
*    'H' = data are from Hipparcos and include proper motion.
*    'T' = data are from Tycho-1. No proper motion is given.
*Note (3):
```

```
*    ' ' = both BT and VT given.
*    'B' = only BT given.
*    'V' = only VT given.
*    'H' = Hp is given instead of VT. BT is blank.
*Note (4):
*    Blank when no magnitude is available.
*    For Hipparcos stars with no VT, Hp is given instead of VT, and BT is blank.
*Note (5):
*    Distance in units of 100 mas to nearest Tycho-2 main or supplement
*    entry. (Computed for the epoch 1991.25). A value of 999 (i.e. 99.9
*    arcsec) is given if the distance exceeds 99.9 arcsec.

c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

C  Declarations for 'index.dat' ! Index to Tycho-2 and supplement-1

      integer*4 nr__3
      parameter (nr__3=9538)    ! Number of records
      character*42 ar__3        ! Full-size record

      integer*4     rec_t2     (nr__3) ! +  Tycho-2 rec. of 1st star in region (1)
      integer*4     rec_s1     (nr__3) ! += Suppl-1 rec. of 1st star in region (1)
      real*4        RAmin      (nr__3) ! (deg) [-0.01,] smallest RA in region (2)
      real*4        RAmax      (nr__3) ! (deg) [,360.00] largest RA in region (2)
      real*4        DEmin      (nr__3) ! (deg) smallest Dec in this region (2)
      real*4        DEmax      (nr__3) ! (deg) largest Dec in this region (2)
*Note (1): The catalogue is sorted according to the GSC region numbers.
*    The line i of the index file gives the record number in Tycho-2 of
*    the first star in GSC region i. Line i+1 gives the record number +1
*    of the last star in GSC region i. For Supplement-1, some regions are
*    empty and line i and line i+1 give the same record number.
*Note (2): a safe rounding was applied. Minimum values are always
*    rounded down and maximum values up.

C===============================================================================

C  Loading file 'tyc2.dat'      ! *The Tycho-2 main catalogue

C  Format for file interpretation

    1 format(
     +  I4,1X,I5,1X,I1,1X,A1,1X,F12.8,1X,F12.8,1X,F7.1,1X,F7.1,1X,I3,
     +  1X,I3,1X,F4.1,1X,F4.1,1X,F7.2,1X,F7.2,1X,I2,1X,F3.1,1X,F3.1,
     +  1X,F3.1,1X,F3.1,1X,F6.3,1X,F5.3,1X,F6.3,1X,F5.3,1X,I3,1X,A1,
     +  1X,I6,A3,1X,F12.8,1X,F12.8,1X,F4.2,1X,F4.2,1X,F5.1,1X,F5.1,1X,
     +  A1,1X,F4.1)

C  Effective file loading

      open(unit=1,status='old',file=
     +'tyc2.dat')
      write(6,*) '....Loading file: tyc2.dat'
      do i__=1,2539913
        read(1,'(A206)')ar__
        read(ar__,1)
     +  TYC1(i__),TYC2(i__),TYC3(i__),pflag(i__),RAmdeg(i__),
     +  DEmdeg(i__),pmRA(i__),pmDE(i__),e_RAmdeg(i__),e_DEmdeg(i__),
     +  e_pmRA(i__),e_pmDE(i__),EpRAm(i__),EpDEm(i__),Num(i__),
     +  q_RAmdeg(i__),q_DEmdeg(i__),q_pmRA(i__),q_pmDE(i__),
     +  BTmag(i__),e_BTmag(i__),VTmag(i__),e_VTmag(i__),prox(i__),
     +  TYC(i__),HIP(i__),CCDM(i__),RAdeg(i__),DEdeg(i__),
     +  EpRA_1990(i__),EpDE_1990(i__),e_RAdeg(i__),e_DEdeg(i__),
     +  posflg(i__),corr(i__)
        if(ar__(16:27) .EQ. '') RAmdeg(i__) = rNULL__
        if(ar__(29:40) .EQ. '') DEmdeg(i__) = rNULL__
        if(ar__(42:48) .EQ. '') pmRA(i__) = rNULL__
        if(ar__(50:56) .EQ. '') pmDE(i__) = rNULL__
        if(ar__(58:60) .EQ. '') e_RAmdeg(i__) = iNULL__
        if(ar__(62:64) .EQ. '') e_DEmdeg(i__) = iNULL__
        if(ar__(66:69) .EQ. '') e_pmRA(i__) = rNULL__
        if(ar__(71:74) .EQ. '') e_pmDE(i__) = rNULL__
        if(ar__(76:82) .EQ. '') EpRAm(i__) = rNULL__
        if(ar__(84:90) .EQ. '') EpDEm(i__) = rNULL__
        if(ar__(92:93) .EQ. '') Num(i__) = iNULL__
        if(ar__(95:97) .EQ. '') q_RAmdeg(i__) = rNULL__
        if(ar__(99:101) .EQ. '') q_DEmdeg(i__) = rNULL__
        if(ar__(103:105) .EQ. '') q_pmRA(i__) = rNULL__
        if(ar__(107:109) .EQ. '') q_pmDE(i__) = rNULL__
        if(ar__(111:116) .EQ. '') BTmag(i__) = rNULL__
        if(ar__(118:122) .EQ. '') e_BTmag(i__) = rNULL__
        if(ar__(124:129) .EQ. '') VTmag(i__) = rNULL__
        if(ar__(131:135) .EQ. '') e_VTmag(i__) = rNULL__
        if(ar__(143:148) .EQ. '') HIP(i__) = iNULL__
c    .............Just test output..........
```

```
          write(6,1)
     +  TYC1(i__),TYC2(i__),TYC3(i__),pflag(i__),RAmdeg(i__),
     +  DEmdeg(i__),pmRA(i__),pmDE(i__),e_RAmdeg(i__),e_DEmdeg(i__),
     +  e_pmRA(i__),e_pmDE(i__),EpRAm(i__),EpDEm(i__),Num(i__),
     +  q_RAmdeg(i__),q_DEmdeg(i__),q_pmRA(i__),q_pmDE(i__),
     +  BTmag(i__),e_BTmag(i__),VTmag(i__),e_VTmag(i__),prox(i__),
     +  TYC(i__),HIP(i__),CCDM(i__),RAdeg(i__),DEdeg(i__),
     +  EpRA_1990(i__),EpDE_1990(i__),e_RAdeg(i__),e_DEdeg(i__),
     +  posflg(i__),corr(i__)
c    .......End.of.Just test output...........
        end do
        close(1)

C===============================================================================

C  Loading file 'suppl_1.dat'   ! The Tycho-2 supplement-1

C  Format for file interpretation

    2 format(
     +  I4,1X,I5,1X,I1,1X,A1,1X,F12.8,1X,F12.8,1X,F7.1,1X,F7.1,1X,
     +  F5.1,1X,F5.1,1X,F5.1,1X,F5.1,1X,A1,1X,F6.3,1X,F5.3,1X,F6.3,1X,
     +  F5.3,1X,I3,1X,A1,1X,I6,A1)

C  Effective file loading

        open(unit=1,status='old',file=
     +'suppl_1.dat')
        write(6,*) '....Loading file: suppl_1.dat'
        do i__=1,17588
          read(1,'(A122)')ar__1
          read(ar__1,2)
     +  TYC1_1(i__),TYC2_1(i__),TYC3_1(i__),flag(i__),RAdeg_1(i__),
     +  DEdeg_1(i__),pmRA_1(i__),pmDE_1(i__),e_RAdeg_1(i__),
     +  e_DEdeg_1(i__),e_pmRA_1(i__),e_pmDE_1(i__),mflag(i__),
     +  BTmag_1(i__),e_BTmag_1(i__),VTmag_1(i__),e_VTmag_1(i__),
     +  prox_1(i__),TYC_1(i__),HIP_1(i__),CCDM_1(i__)
          if(ar__1(42:48) .EQ. '') pmRA_1(i__) = rNULL__
          if(ar__1(50:56) .EQ. '') pmDE_1(i__) = rNULL__
          if(ar__1(70:74) .EQ. '') e_pmRA_1(i__) = rNULL__
          if(ar__1(76:80) .EQ. '') e_pmDE_1(i__) = rNULL__
          if(ar__1(84:89) .EQ. '') BTmag_1(i__) = rNULL__
          if(ar__1(91:95) .EQ. '') e_BTmag_1(i__) = rNULL__
          if(ar__1(97:102) .EQ. '') VTmag_1(i__) = rNULL__
          if(ar__1(104:108) .EQ. '') e_VTmag_1(i__) = rNULL__
          if(ar__1(116:121) .EQ. '') HIP_1(i__) = iNULL__
c    .............Just test output..........
          write(6,2)
     +  TYC1_1(i__),TYC2_1(i__),TYC3_1(i__),flag(i__),RAdeg_1(i__),
     +  DEdeg_1(i__),pmRA_1(i__),pmDE_1(i__),e_RAdeg_1(i__),
     +  e_DEdeg_1(i__),e_pmRA_1(i__),e_pmDE_1(i__),mflag(i__),
     +  BTmag_1(i__),e_BTmag_1(i__),VTmag_1(i__),e_VTmag_1(i__),
     +  prox_1(i__),TYC_1(i__),HIP_1(i__),CCDM_1(i__)
c    .......End.of.Just test output..........
        end do
        close(1)

C===============================================================================

C  Loading file 'suppl_2.dat'   ! The Tycho-2 supplement-2

C  Format for file interpretation

    3 format(
     +  I4,1X,I5,1X,I1,1X,A1,1X,F12.8,1X,F12.8,1X,F7.1,1X,F7.1,1X,
     +  F5.1,1X,F5.1,1X,F5.1,1X,F5.1,1X,A1,1X,F6.3,1X,F5.3,1X,F6.3,1X,
     +  F5.3,1X,I3,1X,A1,1X,I6,A1)

C  Effective file loading

        open(unit=1,status='old',file=
     +'suppl_2.dat')
        write(6,*) '....Loading file: suppl_2.dat'
        do i__=1,1146
          read(1,'(A122)')ar__2
          read(ar__2,3)
     +  TYC1_2(i__),TYC2_2(i__),TYC3_2(i__),flag_1(i__),RAdeg_2(i__),
     +  DEdeg_2(i__),pmRA_2(i__),pmDE_2(i__),e_RAdeg_2(i__),
     +  e_DEdeg_2(i__),e_pmRA_2(i__),e_pmDE_2(i__),mflag_1(i__),
     +  BTmag_2(i__),e_BTmag_2(i__),VTmag_2(i__),e_VTmag_2(i__),
     +  prox_2(i__),TYC_2(i__),HIP_2(i__),CCDM_2(i__)
          if(ar__2(42:48) .EQ. '') pmRA_2(i__) = rNULL__
          if(ar__2(50:56) .EQ. '') pmDE_2(i__) = rNULL__
          if(ar__2(70:74) .EQ. '') e_pmRA_2(i__) = rNULL__
```

```
        if(ar__2(76:80) .EQ. '') e_pmDE_2(i__) = rNULL__
        if(ar__2(84:89) .EQ. '') BTmag_2(i__) = rNULL__
        if(ar__2(91:95) .EQ. '') e_BTmag_2(i__) = rNULL__
        if(ar__2(97:102) .EQ. '') VTmag_2(i__) = rNULL__
        if(ar__2(104:108) .EQ. '') e_VTmag_2(i__) = rNULL__
        if(ar__2(116:121) .EQ. '') HIP_2(i__) = iNULL__
c    .............Just test output..........
        write(6,3)
     +  TYC1_2(i__),TYC2_2(i__),TYC3_2(i__),flag_1(i__),RAdeg_2(i__),
     +  DEdeg_2(i__),pmRA_2(i__),pmDE_2(i__),e_RAdeg_2(i__),
     +  e_DEdeg_2(i__),e_pmRA_2(i__),e_pmDE_2(i__),mflag_1(i__),
     +  BTmag_2(i__),e_BTmag_2(i__),VTmag_2(i__),e_VTmag_2(i__),
     +  prox_2(i__),TYC_2(i__),HIP_2(i__),CCDM_2(i__)
c    .......End.of.Just test output..........
      end do
      close(1)

C===============================================================================

C  Loading file 'index.dat'     ! Index to Tycho-2 and supplement-1

C  Format for file interpretation

    4 format(I7,1X,I6,1X,F6.2,1X,F6.2,1X,F6.2,1X,F6.2)

C  Effective file loading

      open(unit=1,status='old',file=
     +'index.dat')
      write(6,*) '....Loading file: index.dat'
      do i__=1,9538
        read(1,'(A42)')ar__3
        read(ar__3,4)
     +  rec_t2(i__),rec_s1(i__),RAmin(i__),RAmax(i__),DEmin(i__),
     +  DEmax(i__)
c    .............Just test output..........
        write(6,4)
     +  rec_t2(i__),rec_s1(i__),RAmin(i__),RAmax(i__),DEmin(i__),
     +  DEmax(i__)
c    .......End.of.Just test output..........
      end do
      close(1)

C===============================================================================
      stop
      end
```

Small. Fast. Reliable.
Choose any three.

About Documentation Download License Support Purchase

Search SQLite Docs... Go

An Introduction To The SQLite C/C++ Interface

1.0 Executive Summary

The following two objects and eight methods comprise the essential elements of the SQLite interface:

sqlite3	The database connection object. Created by sqlite3_open() and destroyed by sqlite3_close().
sqlite3_stmt	The prepared statement object. Created by sqlite3_prepare() and destroyed by sqlite3_finalize().
sqlite3_open()	Open a connection to a new or existing SQLite database. The constructor for sqlite3.
sqlite3_prepare()	Compile SQL text into byte-code that will do the work of querying or updating the database. The constructor for sqlite3_stmt.
sqlite3_bind()	Store application data into parameters of the original SQL.
sqlite3_step()	Advance an sqlite3_stmt to the next result row or to completion.
sqlite3_column()	Column values in the current result row for an sqlite3_stmt.
sqlite3_finalize()	Destructor for sqlite3_stmt.
sqlite3_close()	Destructor for sqlite3.
sqlite3_exec()	A wrapper function that does sqlite3_prepare(), sqlite3_step(), sqlite3_column(), and sqlite3_finalize() for a string of one or more SQL statements.

2.0 Introduction

SQLite currently has over 200 distinct APIs. This can be overwhelming to a new programmer. Fortunately, most of the interfaces are very specialized and need not be considered by beginners. The core API is small, simple, and easy to learn. This article summarizes the core API.

A separate document, The SQLite C/C++ Interface, provides detailed specifications for all C/C++ APIs for SQLite. Once the reader understands the basic principles of operation for SQLite, that document should be used as a reference guide. This article is intended as introduction only and is neither a complete nor authoritative reference for the SQLite API.

3.0 Core Objects And Interfaces

The principal task of an SQL database engine is to evaluate SQL statements of SQL. To accomplish this, the developer needs two objects:

- The database connection object: sqlite3

- The prepared statement object: sqlite3_stmt

Strictly speaking, the prepared statement object is not required since the convenience wrapper interfaces, sqlite3_exec or sqlite3_get_table, can be used and these convenience wrappers encapsulate and hide the prepared statement object. Nevertheless, an understanding of prepared statements is needed to make full use of SQLite.

The database connection and prepared statement objects are controlled by a small set of C/C++ interface routine listed below.

- sqlite3_open()
- sqlite3_prepare()
- sqlite3_step()
- sqlite3_column()
- sqlite3_finalize()
- sqlite3_close()

Note that the list of routines above is conceptual rather than actual. Many of these routines come in multiple versions. For example, the list above shows a single routine named sqlite3_open() when in fact there are three separate routines that accomplish the same thing in slightly different ways: sqlite3_open(), sqlite3_open16() and sqlite3_open_v2(). The list mentions sqlite3_column() when in fact no such routine exists. The "sqlite3_column()" shown in the list is place holders for an entire family of routines to be used for extracting column data in various datatypes.

Here is a summary of what the core interfaces do:

sqlite3_open()

This routine opens a connection to an SQLite database file and returns a database connection object. This is often the first SQLite API call that an application makes and is a prerequisite for most other SQLite APIs. Many SQLite interfaces require a pointer to the database connection object as their first parameter and can be thought of as methods on the database connection object. This routine is the constructor for the database connection object.

sqlite3_prepare()

This routine converts SQL text into a prepared statement object and returns a pointer to that object. This interface requires a database connection pointer created by a prior call to sqlite3_open() and a text string containing the SQL statement to be prepared. This API does not actually evaluate the SQL statement. It merely prepares the SQL statement for evaluation.

Think of each SQL statement as a small computer program. The purpose of sqlite3_prepare() is to compile that program into object code. The prepared statement is the object code. The sqlite3_step() interface then runs the object code to get a result.

New applications should always invoke sqlite3_prepare_v2() instead of sqlite3_prepare(). The older sqlite3_prepare() is retained for backwards compatibility. But sqlite3_prepare_v2() provides a much better interface.

sqlite3_step()

This routine is used to evaluate a prepared statement that has been previously created by the sqlite3_prepare() interface. The statement is evaluated up to the point where the first row of results are available. To advance to the second row of results, invoke sqlite3_step() again. Continue invoking sqlite3_step() until the statement is complete. Statements that do not return results (ex: INSERT, UPDATE, or DELETE statements) run to completion on a single call to sqlite3_step().

sqlite3_column() This routine returns a single column from the current row of a result set for a prepared statement that is being evaluated by sqlite3_step(). Each time sqlite3_step() stops with a new result set row, this routine can be called multiple times to find the values of all columns in that row.

As noted above, there really is no such thing as a "sqlite3_column()" function in the SQLite API. Instead, what we here call "sqlite3_column()" is a place-holder for an entire family of functions that return a value from the result set in various data types. There are also routines in this family that return the size of the result (if it is a string or BLOB) and the number of columns in the result set.

- sqlite3_column_blob()
- sqlite3_column_bytes()
- sqlite3_column_bytes16()
- sqlite3_column_count()
- sqlite3_column_double()
- sqlite3_column_int()
- sqlite3_column_int64()
- sqlite3_column_text()
- sqlite3_column_text16()
- sqlite3_column_type()
- sqlite3_column_value()

sqlite3_finalize() This routine destroys a prepared statement created by a prior call to sqlite3_prepare(). Every prepared statement must be destroyed using a call to this routine in order to avoid memory leaks.

sqlite3_close() This routine closes a database connection previously opened by a call to sqlite3_open(). All prepared statements associated with the connection should be finalized prior to closing the connection.

3.1 Typical Usage Of Core Routines And Objects

An application will typically use sqlite3_open() to create a single database connection during initialization. Note that sqlite3_open() can be used to either open existing database files or to create and open new database files. While many applications use only a single database connection, there is no reason why an application cannot call sqlite3_open() multiple times in order to open multiple database connections - either to the same database or to different databases. Sometimes a multi-threaded application will create separate database connections for each threads. Note that a single database connection can access two or more databases using the ATTACH SQL command, so it is not necessary to have a separate database connection for each database file.

Many applications destroy their database connections using calls to sqlite3_close() at shutdown. Or, for example, an application that uses SQLite as its application file format might open database connections in response to a File/Open menu action and then destroy the corresponding database connection in response to the File/Close menu.

To run an SQL statement, the application follows these steps:

1. Create a prepared statement using sqlite3_prepare().
2. Evaluate the prepared statement by calling sqlite3_step() one or more times.
3. For queries, extract results by calling sqlite3_column() in between two calls to sqlite3_step().
4. Destroy the prepared statement using sqlite3_finalize().

The foregoing is all one really needs to know in order to use SQLite effectively. All the rest is optimization and detail.

4.0 Convenience Wrappers Around Core Routines

The sqlite3_exec() interface is a convenience wrapper that carries out all four of the above steps with a single function call. A callback function passed into sqlite3_exec() is used to process each row of the result set. The sqlite3_get_table() is another convenience wrapper that does all four of the above steps. The sqlite3_get_table() interface differs from sqlite3_exec() in that it stores the results of queries in heap memory rather than invoking a callback.

It is important to realize that neither sqlite3_exec() nor sqlite3_get_table() do anything that cannot be accomplished using the core routines. In fact, these wrappers are implemented purely in terms of the core routines.

5.0 Binding Parameters and Reusing Prepared Statements

In prior discussion, it was assumed that each SQL statement is prepared once, evaluated, then destroyed. However, SQLite allows the same prepared statement to be evaluated multiple times. This is accomplished using the following routines:

- sqlite3_reset()
- sqlite3_bind()

After a prepared statement has been evaluated by one or more calls to sqlite3_step(), it can be reset in order to be evaluated again by a call to sqlite3_reset(). Think of sqlite3_reset() as rewinding the prepared statement program back to the beginning. Using sqlite3_reset() on an existing prepared statement rather than creating a new prepared statement avoids unnecessary calls to sqlite3_prepare(). For many SQL statements, the time needed to run sqlite3_prepare() equals or exceeds the time needed by sqlite3_step(). So avoiding calls to sqlite3_prepare() can give a significant performance improvement.

It is not commonly useful to evaluate the *exact* same SQL statement more than once. More often, one wants to evaluate similar statements. For example, you might want to evaluate an INSERT statement multiple times with different values. Or you might want to evaluate the same query multiple times using a different key in the WHERE clause. To accommodate this, SQLite allows SQL statements to contain parameters which are "bound" to values prior to being evaluated. These values can later be changed and the same prepared statement can be evaluated a second time using the new values.

SQLite allows a parameter wherever a string literal, numeric constant, or NULL is allowed. (Parameters may not be used for column or table names.) A parameter takes one of the following forms:

- **?**
- **?***NNN*
- **:***AAA*
- **$***AAA*
- *@AAA*

In the examples above, *NNN* is an integer value and *AAA* is an identifier. A parameter initially has a value of NULL. Prior to calling sqlite3_step() for the first time or immediately after sqlite3_reset(), the application can invoke the sqlite3_bind() interfaces to attach values to the parameters. Each call to sqlite3_bind() overrides prior bindings on the same parameter.

An application is allowed to prepare multiple SQL statements in advance and evaluate them as needed. There is no arbitrary limit to the number of outstanding prepared statements. Some applications call sqlite3_prepare() multiple times at start-up to create all of the prepared statements they will ever need. Other applications keep a cache of the most recently used prepared statements and then reuse prepared statements out of the cache when available. Another approach is to only reuse prepared statements when they are inside of a loop.

6.0 Configuring SQLite

The default configuration for SQLite works great for most applications. But sometimes developers want to tweak the setup to try to squeeze out a little more performance, or take advantage of some obscure feature.

The sqlite3_config() interface is used to make global, process-wide configuration changes for SQLite. The sqlite3_config() interface must be called before any database connections are created. The sqlite3_config() interface allows the programmer to do things like:

- Adjust how SQLite does memory allocation, including setting up alternative memory allocators appropriate for safety-critical real-time embedded systems and application-defined memory allocators.
- Set up a process-wide error log.
- Specify an application-defined page cache.
- Adjust the use of mutexes so that they are appropriate for various threading models, or substitute an application-defined mutex system.

After process-wide configuration is complete and database connections have been created, individual database connections can be configured using calls to sqlite3_limit() and sqlite3_db_config().

7.0 Extending SQLite

SQLite includes interfaces that can be used to extend its functionality. Such routines include:

- sqlite3_create_collation()
- sqlite3_create_function()
- sqlite3_create_module()
- sqlite3_vfs_register()

The sqlite3_create_collation() interface is used to create new collating sequences for sorting text. The sqlite3_create_module() interface is used to register new virtual table implementations. The sqlite3_vfs_register() interface creates new VFSes.

The sqlite3_create_function() interface creates new SQL functions - either scalar or aggregate. The new function implementation typically makes use of the following additional interfaces:

- sqlite3_aggregate_context()
- sqlite3_result()
- sqlite3_user_data()
- sqlite3_value()

All of the built-in SQL functions of SQLite are created using exactly these same interfaces. Refer to the SQLite source code, and in particular the date.c and func.c source files for examples.

Shared libraries or DLLs can be used as loadable extensions to SQLite.

8.0 Other Interfaces

This article only mentions the most important and most commonly used SQLite interfaces. The SQLite library includes many other APIs implementing useful features that are not described here. A complete list of functions that form the SQLite application programming interface is found at the C/C++ Interface Specification. Refer to that document for complete and authoritative information about all SQLite interfaces.

UBV photometric system

From Wikipedia, the free encyclopedia

The **UBV photometric system**, also called the Johnson system (or Johnson-Morgan system), is a wide band photometric system for classifying stars according to their colors. It is the first known standardized photoelectric photometric system. The letters U, B, and V stand for ultraviolet, blue, and visual magnitudes, which are measured for a star in order to classify it in the UBV system.[1] The choice of colors on the blue end of the spectrum is because of the bias that photographic film has for those colors. It was introduced in the 1950s by American astronomers Harold Lester Johnson and William Wilson Morgan. A 13" telescope and the 82" telescope at McDonald Observatory were used to define the system.[1][2]

The filters are selected so that the mean wavelengths of response functions are 364 nm for U, 442 nm for B, 540 nm for V. The zero point of the B−V and U−B color indices were defined such as to be about zero for A0 main sequence stars not affected by interstellar reddening.[1]

The UBV system has some disadvantages. The short wavelength cutoff of the U filter is defined mainly by the terrestrial atmosphere rather than the filter itself; thus, it (and observed magnitudes) can vary with altitude and atmospheric conditions.[3] However, a large number of measurements have been made in this system, including many of the bright stars.[4]

Extensions

The Johnson-Cousins UBVRI photometric system is a common extension of Johnson's original system that provides redder passbands.

See also

- Photometric system
- Stellar classification

References

1. Johnson, H. L.; Morgan, W. W. (1953). "Fundamental stellar photometry for standards of spectral type on the revised system of the Yerkes spectral atlas". *The Astrophysical Journal* **117** (3): 313–352. Bibcode:1953ApJ...117..313J. doi:10.1086/145697.
2. Hearnshaw, J. B. (1996). *The Measurement of Starlight: Two Centuries of Astronomical Photometry*. Cambridge University Press. p. 421. ISBN 9780521403931. "In the 1950–51 winter, Johnson had commenced photometry in three passbands (designated U, V, and Y) on the McDonald 13- and 82-inch reflectors [54]."
3. Hearnshaw, J. B. (1996). *The Measurement of Starlight: Two Centuries of Astronomical Photometry*. Cambridge University Press. p. 425. ISBN 9780521403931.
4. Iriarte, Braulio, Johnson, Harold L., Mitchell, Richard I., and Wisniewski, Wieslaw K. (1965), *Five-Color Photometry of Bright Stars*, Sky & Telescope, vol. 30, p. 21

Visible spectrum

From Wikipedia, the free encyclopedia

The **visible spectrum** is the portion of the electromagnetic spectrum that is visible to the human eye. Electromagnetic radiation in this range of wavelengths is called **visible light** or simply light. A typical human eye will respond to wavelengths from about 390 to 700 nm.[1] In terms of frequency, this corresponds to a band in the vicinity of 430–770 THz.

White light is dispersed by a prism into the colors of the visible spectrum.

The spectrum does not, however, contain all the colors that the human eyes and brain can distinguish. Unsaturated colors such as pink, or purple variations such as magenta, are absent, for example, because they can be made only by a mix of multiple wavelengths. Colors containing only one wavelength are also called pure colors or spectral colors.

Visible wavelengths pass through the "optical window", the region of the electromagnetic spectrum that allows wavelengths to pass largely unattenuated through the Earth's atmosphere. An example of this phenomenon is that clean air scatters blue light more than red wavelengths, and so the midday sky appears blue. The optical window is also referred to as the "visible window" because it overlaps the human visible response spectrum. The near infrared (NIR) window lies just out of the human vision, as well as the Medium Wavelength IR (MWIR) window, and the Long Wavelength or Far Infrared (LWIR or FIR) window, although other animals may experience them.

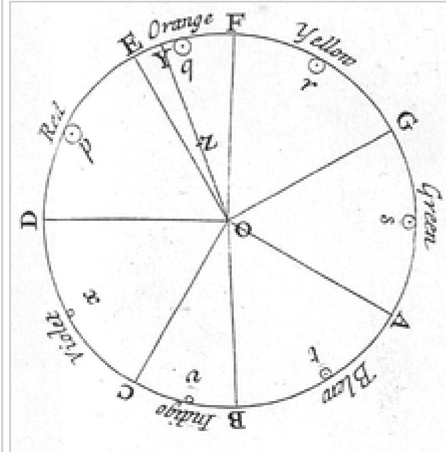
Contents

History

In the 13th century, Roger Bacon theorized that rainbows were produced by a similar process to the passage of light through glass or crystal.[2]

In the 17th century, Isaac Newton discovered that prisms could disassemble and reassemble white light, and described the phenomenon in his book *Opticks*. He was the first to use the word *spectrum* (Latin for "appearance" or "apparition") in this sense in print in 1671 in describing his experiments in optics. Newton observed that, when a narrow beam of sunlight strikes the face of a glass prism at an angle, some is reflected and some of the beam passes into and through the glass, emerging as different-colored bands. Newton hypothesized light to be made up of "corpuscles" (particles) of different colors, with the different colors of light moving at different speeds in transparent matter, red light moving more quickly than violet in glass. The result is that red light is bent (refracted) less sharply than violet as it passes through the prism, creating a spectrum of colors.

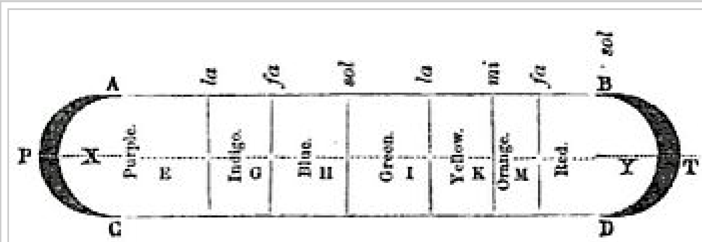
Newton divided the spectrum into seven named colors: red, orange, yellow, green, blue, indigo, and violet. He chose seven colors out of a belief, derived from the ancient Greek sophists, of there being a connection between the colors, the musical notes, the known objects in the solar system, and the days of the week.[3][4] The human eye is relatively insensitive to indigo's frequencies, and some people who have otherwise-good vision cannot distinguish indigo from blue and violet. For this reason, some later commentators, including Isaac Asimov,[5] have suggested that indigo should not be regarded as a color in its own right but merely as a shade of blue or violet. However, the evidence indicates that what Newton meant by "indigo" and "blue" does not correspond to the modern meanings of those color words. Comparing Newton's observation of prismatic colors to a color image of the visible light spectrum shows that "indigo" corresponds to what is today called blue, whereas "blue" corresponds to cyan.[6][7][8]



Newton's color circle, from *Opticks* of 1704, showing the colors correlated with musical notes. The spectral colors from red to violet are divided by the notes of the musical scale, starting at D. The circle completes a full octave, from D to D. Newton's circle places red, at one end of the spectrum, next to violet, at the other. This reflects the fact that non-spectral purple colors are observed when red and violet light are mixed.

In the 18th century, Goethe wrote about optical spectra in his *Theory of Colours*. Goethe used the word *spectrum* (*Spektrum*) to designate a ghostly optical afterimage, as did Schopenhauer in *On Vision and Colors*. Goethe argued that the continuous spectrum was a compound phenomenon. Where Newton narrowed the beam of light to isolate the phenomenon, Goethe observed that a wider aperture produces not a spectrum but rather reddish-yellow and blue-cyan edges with white between them. The spectrum appears only when these edges are close enough to overlap.

In the early 19th century, the concept of the visible spectrum became more definite, as light outside the visible range was discovered and characterized by William Herschel (infrared) and Johann Wilhelm Ritter (ultraviolet), Thomas Young, Thomas Johann Seebeck, and others.[9] Young was the first to measure the wavelengths of different colors of light, in 1802.[10]



Newton's observation of prismatic colors (David Brewster 1855)

The connection between the visible spectrum and color vision was explored by Thomas Young and Hermann von Helmholtz in the early 19th century. Their theory of color vision correctly proposed that the eye uses three distinct receptors to perceive color.

Animal color vision

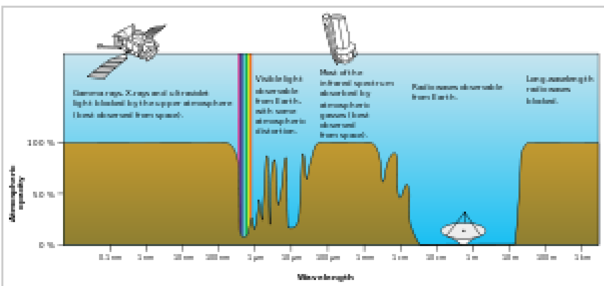
Many species can see light with frequencies outside the human "visible spectrum". Bees and many other insects can detect ultraviolet light, which helps them find nectar in flowers. Plant species that depend on insect pollination may owe reproductive success to their appearance in ultraviolet light rather than how colorful they appear to humans. Birds, too, can see into the ultraviolet (300–400 nm), and some have sex-dependent markings on their plumage that are visible only in the ultraviolet range.[11][12] Many animals that can see into the ultraviolet range, however, cannot see red light or any other reddish wavelengths. Bees' visible spectrum ends at about 590 nm, just before the orange wavelengths start.[13] Birds, however, can see some red wavelengths, although not as far into the light spectrum as humans.[14] The popular belief that the common goldfish is the only animal that can see both infrared and ultraviolet light [15] is incorrect, because goldfish cannot see infrared light.[16] Similarly, dogs are often thought to be color blind but they have been shown to be sensitive to colors, though not as many as humans.[17]

Spectral colors

Colors that can be produced by visible light of a narrow band of wavelengths (monochromatic light) are called pure spectral colors. The various color ranges indicated in the illustration are an approximation: The spectrum is continuous, with no clear boundaries between one color and the next.[18]

Color	Wavelength	Frequency	Photon energy
violet	380–450 nm	668–789 THz	2.75–3.26 eV
blue	450–495 nm	606–668 THz	2.50–2.75 eV
green	495–570 nm	526–606 THz	2.17–2.50 eV
yellow	570–590 nm	508–526 THz	2.10–2.17 eV
orange	590–620 nm	484–508 THz	2.00–2.10 eV
red	620–750 nm	400–484 THz	1.65–2.00 eV

Spectroscopy



Rough plot of Earth's atmospheric opacity to various wavelengths of electromagnetic radiation, including visible light

Spectroscopy is the study of objects based on the spectrum of color they emit, absorb or reflect. Spectroscopy is an important investigative tool in astronomy, where scientists use it to analyze the properties of distant objects. Typically, astronomical spectroscopy uses high-dispersion diffraction gratings to observe spectra at very high spectral resolutions. Helium was first detected by analysis of the spectrum of the sun. Chemical elements can be detected in astronomical objects by emission lines and absorption lines.

The shifting of spectral lines can be used to measure the Doppler shift (red shift or blue shift) of distant objects.

Color display spectrum

Color displays (e.g. computer monitors and televisions) cannot reproduce *all* colors discernible by a human eye. Colors outside the color gamut of the device, such as most spectral colors, can only be approximated. For color-accurate reproduction, a spectrum can be projected onto a uniform gray field. The resulting mixed colors can have all their R,G,B coordinates non-negative, and so can be reproduced without distortion. This accurately simulates looking at a spectrum on a gray background.[19]

Approximation of spectral colors on a display results in somewhat distorted chromaticity

A rendering of the visible spectrum on a gray background produces non-spectral mixtures of pure spectrum with gray, which fit into the sRGB color space.

See also

- High-energy visible light

Wikisource has original text related to this article: **Definition of the Color Indigo**

Wikimedia Commons has media related to ***Visible spectrum***.

References

1. Cecie Starr (2005). *Biology: Concepts and Applications*. Thomson Brooks/Cole. ISBN 0-534-46226-X.
2. Coffey, Peter (1912). *The Science of Logic: An Inquiry Into the Principles of Accurate Thought*. Longmans.
3. Isacoff, Stuart (16 January 2009). *Temperament: How Music Became a Battleground for the Great Minds of Western Civilization*. Knopf Doubleday Publishing Group. pp. 12–13. ISBN 978-0-307-56051-3. Retrieved 18 March 2014.
4. Brebbia, C. A.; Greated, C.; Collins, M. W. (1 January 2011). *Colour in Art, Design & Nature*. WIT Press. pp. 52–3. ISBN 978-1-84564-568-7. Retrieved 18 March 2014.
5. Asimov, Isaac (1975). *Eyes on the universe : a history of the telescope*. Boston: Houghton Mifflin. p. 59. ISBN 978-0-395-20716-1.
6. Evans, Ralph M. (1974). *The perception of color.* (null ed.). New York: Wiley-Interscience. ISBN 978-0-471-24785-2.
7. McLaren, K. (March 2007). "Newton's indigo". *Color Research & Application* **10** (4): 225–229. doi:10.1002/col.5080100411.
8. Waldman, Gary (2002). *Introduction to light : the physics of light, vision, and color* (Dover ed.). Mineola: Dover Publications. p. 193. ISBN 978-0-486-42118-6.
9. Mary Jo Nye (editor) (2003). *The Cambridge History of Science: The Modern Physical and Mathematical Sciences* **5**. Cambridge University Press. p. 278. ISBN 978-0-521-57199-9.
10. John C. D. Brand (1995). *Lines of light: the sources of dispersive spectroscopy, 1800–1930*. CRC Press. pp. 30–32. ISBN 978-2-88449-163-1.
11. Cuthill, Innes C (1997). "Ultraviolet vision in birds". In Peter J.B. Slater. *Advances in the Study of Behavior* **29**. Oxford, England: Academic Press. p. 161. ISBN 978-0-12-004529-7.
12. Jamieson, Barrie G. M. (2007). *Reproductive Biology and Phylogeny of Birds*. Charlottesville VA: University of Virginia. p. 128. ISBN 1-57808-386-9.
13. Skorupski, Peter; Chittka, Lars (10 August 2010). "Photoreceptor Spectral Sensitivity in the Bumblebee, Bombus impatiens (Hymenoptera: Apidae)". *PLoS ONE* **5** (8): e12049. Bibcode:2010PLoSO...512049S. doi:10.1371/journal.pone.0012049.
14. Varela, F. J.; Palacios, A. G.; Goldsmith T. M. "Color vision of birds" (https://books.google.com/books/about/Vision_Brain_and_Behavior_in_Birds.html?id=p1SUzc5GUVcC&redir_esc=y) in Ziegler & Bischof (1993) 77–94
15. "True or False? "The common goldfish is the only animal that can see both infra-red and ultra-violet light." – Skeptive". Retrieved September 28, 2013.
16. Neumeyer, Christa (2012). "Chapter 2: Color Vision in Goldfish and Other Vertebrates". In Lazareva, Olga; Shimizu, Toru; Wasserman, Edward. *How Animals See the World: Comparative Behavior, Biology, and Evolution of Vision*. Oxford Scholarship Online. ISBN 978-0-19-533465-4.
17. "Colour cues proved to be more informative for dogs than brightness". Retrieved November 29, 2015.
18. Thomas J. Bruno, Paris D. N. Svoronos. *CRC Handbook of Fundamental Spectroscopic Correlation Charts. (https://books.google.com/books?id=FgjHjhCh5wsC&pg=PP1&dq=intitle:%22CRC+Handbook+of+Fundamental+Spectroscopic+Correlation+Charts%22&ei=A3TYRvGjJYqKoQK5oYzMBQ&sig=rsr8R_QF8j-fcWljMbTPF14Kcms#PPA2,M1)* CRC Press, 2005.
19. "Reproducing Visible Spectra". *RepairFAQ.org*. Retrieved 2011-02-09.

Retrieved from "https://en.wikipedia.org/w/index.php?title=Visible_spectrum&oldid=720526851"

Categories: Color | Electromagnetic spectrum | Optical spectrum | Vision