List of all Fortran-Programs in alphabetical order	
Name of the Program	Signature
A1111LOAD.FOR	For-11
AARILDGOLOAD.FOR	For-19
AARILOAD.FOR	For-20
ARGENTINE.FOR	For-15
ARGENTINELOAD.FOR	For-21
ARGNEWCRNUM.FOR	For-24
AWILOAD.FOR	For-18
BSHLOAD.FOR	For-17
BSH2LOAD.FOR	For-26
GONELLA.FOR	For-9
GORDON.FOR	For-2
HAINESLOAD.FOR	For-8
KUROPATKIN.FOR	For-7
JARELOAD.FOR	For-13
MARS.FOR	For-5
MARSPROB.FOR	For-10
MUENCHLOAD.FOR	For-14
NOWLIN.FOR	For-6
OZEDB_SYBASE.FOR	For-1
OZEDB_SYBASE1.FOR	For-3
OZEDB_SYBASE2.FOR	For-4
READARGENT.FOR	For-25
READJARE.FOR	For-22
READMUIN.FOR	For-23
REIDINTLOAD.FOR	For-27
REIDOBSLOAD.FOR	For-28
SCHLITZERLOAD.FOR	For-16
TOKYOLOAD.FOR	For-12

List of all Fortran-Programs in numerical order	
Name of the Program	Signature
OZEDB_SYBASE.FOR	For-1
GORDON.FOR	For-2
OZEDB_SYBASE1.FOR	For-3
OZEDB_SYBASE2.FOR	For-4
MARS.FOR	For-5
NOWLIN.FOR	For-6
KUROPATKIN.FOR	For-7
HAINESLOAD.FOR	For-8
GONELLA.FOR	For-9
MARSPROB.FOR	For-10
A1111LOAD.FOR	For-11
TOKYOLOAD.FOR	For-12
JARELOAD.FOR	For-13
MUENCHLOAD.FOR	For-14
ARGENTINE.FOR	For-15
SCHLITZERLOAD.FOR	For-16
BSHLOAD.FOR	For-17
AWILOAD.FOR	For-18
AARILDGOLOAD.FOR	For-19
AARILOAD.FOR	For-20
ARGENTINELOAD.FOR	For-21
READJARE.FOR	For-22
READMUIN.FOR	For-23
ARGNEWCRNUM.FOR	For-24
READARGENT.FOR	For-25
BSH2LOAD.FOR	For-26
REIDINTLOAD.FOR	For-27
REIDOBSLOAD.FOR	For-28

OzE_DB_Sybase.FOR
10.7.89

```
      options /check=all
      program ozedb_load

C     CREATOR::M. Reinke
C     CREA_DATE::10-Jul-1989
      structure /station/
      integer *4        ID
      integer *4        CRUISE_NUMBER
      integer *4        STATION_NUMBER
      real *8           LATITUDE
      real *8           LONGITUDE
      integer *4        BOTTOM_DEPTH
      integer *4        MAX_OBSE_DEPTH
      integer *4        NUMBER_OBSE
      integer *4        MARSDEN_SQUARE
      end structure


      structure /data/
      integer*4         ID
      integer*4         AARI_Station_ID
      real*8            TEMPERATURE
      real*8            SALINITY
      real*8            OXYGEN
      integer*4         DEPTH
      end structure


    include '(fsybdb)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                  err_handler
      EXTERNAL                  msg_handler

      INTEGER*4                 login
      INTEGER*4                 dbproc
      INTEGER*4                 return_code


      INTEGER*4                 error
      CHARACTER*(256)           cmdbuf


        record /station/ station
        record /data/ data


        CHARACTER*15 COM_STRING

        DIMENSION A(12),T(42),S(42),OX(42),Z(42)
        INTEGER*2 A,T,S,OX,Z,leap_year

        INTEGER ID_STAT,ID_DATA,STATUS,LUN
        CHARACTER*30 ASCII_TIME
        CHARACTER*4 JAHR
        CHARACTER*2 TAG
        CHARACTER*2 STUNDE
        INTEGER MONAT
        CHARACTER*3 MONTH(12)
```

Lade prog.:

[OZEDS.DATA]

DISK2.DAT

```
c       ******these are the standard levels depths:
        DATA Z / 0, 10, 20, 30, 50, 75, 100, 125, 150, 200, 250,
     *           300, 350, 400, 500, 600, 700, 750, 800, 900,
     *           1000, 1100, 1200, 1300, 1400, 1500, 1750, 2000,
     *           2250, 2500, 2750, 3000, 3250, 3500, 3750, 4000,
     *           4500, 5000, 5500, 6000, 6500, 7000 /
c
c        a(1) - archiv number of cruise
c        a(2) - cruise number of station
c        a(3) - latitude (in degrees * 100)
c        a(4) - longitude (in degrees * 100)
c        a(5) - year
c        a(6) - month
c        a(7) - day
c        a(8) - hour
c        a(9) - bottom depth (?)
c        a(10) - depth of the deepest observed level
c        a(11) - total number of observed levels
c        a(12) - Marsden square
c
c        t - array of interpolated temperature values ( * 1000 )
c        s - array of interpolated salinity values ( ( S - 30 ) * 1000 )
c        ox - array of interpolated oxygen values ( * 100 )
c
c

        DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
        2           'Sep','Oct','Nov','Dec'/



C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C

      login = fdblogin()
      call fdbsetluser(login, 'REINKE')
      call fdbsetlpwd(login, 'ihlea')
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)

        call fdbuse(dbproc,'SouthernOceanDB')

c       *****this program read the interpolated data from the disk


        OPEN(LUN,FILE='OTH$DATEN:[VGURETS]DISK2.DAT',
        1STATUS='OLD', ACCESS='SEQUENTIAL',
        1RECL=276, FORM='FORMATTED',RECORDTYPE='FIXED')
c
C       *****Zaehlung der  Records
        ID_STAT=0
        ID_DATA=0
```

```
C       *****Eroeffnen der Transaktion


10      CONTINUE
C       *****Lesen des Files
        READ(LUN,100,END=3)A,T,S,OX
100     FORMAT(138A2)


C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(a(5),4)

        if ((a(8).gt.24 .or. a(8) .lt. 00) .OR.
        1   (a(7).gt.31 .or. a(7) .lt. 1 ) .OR.
        1   (a(6).gt.12 .or. a(6) .lt. 1)  .OR.
        1   (a(5).gt.1989 .or. a(5) .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'


C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (a(7).eq.29 .and.
        1        a(6).eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'

        ELSE



         WRITE (TAG,'(I2)') A(7)
         WRITE (JAHR,'(I4)') A(5)
         IF (a(8) .eq. 24) THEN
          Stunde ='23'
         ELSE
          WRITE (STUNDE,'(I2)') A(8)
         END IF
         MONAT=A(6)
        END IF

        ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':00'//'"'

C       ***Speicherung der Stationsdaten****************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=A(1)
        STATION.STATION_NUMBER=A(2)
        STATION.LATITUDE=DFLOAT(A(3))/100.
        STATION.LONGITUDE=DFLOAT(A(4))/100.
        STATION.BOTTOM_DEPTH=A(9)
        STATION.MAX_OBSE_DEPTH=A(10)
        STATION.NUMBER_OBSE=A(11)
        STATION.MARSDEN_SQUARE=A(12)
```

```
      type *, station.id,' ',ascii_time

      call fdbcmd(dbproc,' insert into Aari_Station values ( ')
      call fdbfcmd(dbproc,' %d,', STATION.ID)
      call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
      call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
      call fdbfcmd(dbproc,' %s,', ASCII_TIME)
      call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
      call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
      call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
      call fdbfcmd(dbproc,' %d,', STATION.MARSDEN_SQUARE)
      call fdbcmd(dbproc,'0,0)')

    call fdbsqlexec(dbproc)
    return_code = fdbresults(dbproc)

C     ********Speicherung der Messdaten*******
      DO I=1,42
      ID_DATA=ID_DATA+1
      DATA.ID=ID_DATA
      DATA.AARI_STATION_ID=ID_STAT
      DATA.DEPTH=Z(I)
      IF (T(I).NE.-9999) THEN
       DATA.TEMPERATURE=DFLOAT(T(I))/1000.
      ELSE
       DATA.TEMPERATURE=DFLOAT(T(I))
      END IF
      IF(S(I).NE.-9999) THEN
       DATA.SALINITY=DFLOAT(S(I))/1000. +30.
      ELSE
       DATA.SALINITY=DFLOAT(S(I))
      END IF

      IF (OX(I).NE.-9999) THEN
       DATA.OXYGEN=DFLOAT(OX(I))/100.
      ELSE
       DATA.OXYGEN=DFLOAT(OX(I))
      END IF

      IF ( .NOT.((DATA.TEMPERATURE).EQ.-9999. .AND.
     1           (DATA.SALINITY)    .EQ.-9999. .AND.
     1           (DATA.OXYGEN)      .EQ.-9999.)) THEN
      call fdbcmd(dbproc,' insert into Aari_Standard_Data values (')
      call fdbfcmd(dbproc,' %d,', DATA.ID)
      call fdbfcmd(dbproc,' %d,', DATA.AARI_STATION_ID)
      call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
      call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
      call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
      call fdbfcmd(dbproc,' %f,', DATA.OXYGEN )
      call fdbcmd(dbproc,' 0,0)')

    call fdbsqlexec(dbproc)
    return_code = fdbresults(dbproc)
      END IF


      END DO




      GOTO 10
3     CONTINUE
```

```
        TYPE *,'end of file'
        TYPE *,' there are ', i6, ' stations in the file' ,ISTAT
        CLOSE(LUN)
        call fdbexit()
        END

C
C     ERR_HANDLER - This funtion may be coded within the same program
C     or as a separate file that is compiled/linked.
C

      INTEGER*4 FUNCTION err_handler (dbproc, severity, errno, oserrno)
C
      include '(fsybdb)'
C
      INTEGER*4       dbproc
      INTEGER*4       severity
      INTEGER*4       errno
      INTEGER*4       oserrno
      INTEGER*4       length
      INTEGER*4       return_code
C
      CHARACTER*(80)  message
C
         length = fdberrstr(errno,message)
         type *, 'DB-LIBRARY error: ', message
C
C     Check for operating system errors
C
         length = 0
         message = ' '
         length = fdboserrstr(oserrno, message)
C
         if (oserrno .ne. DBNOERR) then
            type *, 'Operating-system error: ', message
         end if
C
         return_code = fdbdead(dbproc)
C
         if ((dbproc .eq. NULL) .OR. (return_code ) .OR.
     2       (severity .eq. EXSERVER)) then
            err_handler = INT_EXIT
C
         else
            err_handler = INT_CANCEL
         end if
C
         END
C
C MSG_HANDLER - This funtion may be coded within the same program
C               or as a separate file that is compiled/linked.
C

      INTEGER*4 FUNCTION msg_handler (dbproc, msgno,
     2            msgstate,severity, msgtext)
C
      include '(fsybdb)'
C
      INTEGER*4       dbproc
      INTEGER*4       msgno
      INTEGER*4       msgstate
      INTEGER*4       severity
C
      CHARACTER*80    msgtext
        IF (MSGNO.NE.5701) THEN
C
```

```
        type *, 'DataServer message ', msgno,
     2       '   state ', msgstate, '    severity ',
     3      severity,' ', msgtext
C
        END IF
        msg_handler = DBNOSAVE

      END
```

~~ote_sybase.f~~

Gordon Prr, 43
17.7.89

```
      options /check=all
      program gordon_Load

C     CREATOR::M. Reinke
C     CREA_DATE::17-Jul-1989


      structure /gstation/
      character*11     SQUARE_STRING
      character*1      LATI_NAME
      integer*4        LATI,
      1                LATI_MIN
      character*1      LONG_NAME
      character*3      LONG
      character*3      LONG_MIN
      integer*4        YEAR,
      1                MONTH,
      1                DAY,
      1                HOUR
      character*6      SHIP_NAME
      integer*4        BOTTOM_DEPTH,
      1                MIN_OBSE_DEPTH,
      1                MAX_OBSE_DEPTH
      character*4      DIFF_DEPTH
      character*8      OPTION
      character*47     ETC_STRING
      integer          NUMBER_OBSE,
      1                NUMBER_STD_DEPTH


      end structure

      structure /GDATA/
      integer*4        DEPTH,
      1       DEPTH_QUAL,
      1       TEMP,
      1       TEMP_PREC,
      1       TEMP_QUAL,
      1       SAL,
      1       SAL_PREC,
      1       SAL_QUAL,
      1       SIGMA,
      1       SIGMA_QUAL,
      1       SOUND_VEL,
      1       SOUND_VEL_PREC,
      1       OXYGEN,
      1       OXYGEN_PREC,
      1       OXYGEN_QUAL,
      1       IPO4,
      1       IPO4_PREC,
      1       TOP4,
      1       TPO4_PREC,
      1       SIO4,
      1       SIO4_PREC,
      1       NO2,
      1       NO2_PREC,
      1       NO3,
      1       NO3_PREC,
      1       PH,
      1       PH_PREC
      CHARACTER*3 OBSE_FLAG
      end structure
```

[ORDR. GORDON]

FORDOR. DAT

→ GORDON_STANDARD_DATA

```
      structure /station/
      integer *4        ID
      CHARACTER*6       SHIP_NAME
      real *8           LATITUDE
      real *8           LONGITUDE
      integer *4        BOTTOM_DEPTH
      integer *4        MAX_OBSE_DEPTH
      integer *4        NUMBER_OBSE
      integer *4        MARSDEN_SQUARE
      end structure


      structure /data/
      integer*4         ID
      integer*4         GORDON_STATION_ID
      real*8            TEMPERATURE
      real*8            SALINITY
      real*8            OXYGEN
      real*8            IPO4
      real*8            SIO4
      real*8            NO3
      integer*4         DEPTH
      end structure


      structure /stat/
      integer*4         t_num,
      1                 s_num,
      1                 o2_num,
      1                 ipo4_num,
      1                 tpo4_num,
      1                 sio4_num,
      1                 no3_num,
      1                 no4_num,
      1                 ph_num
      integer*4         quality_flag
      integer*4         sigma_num
      integer*4         sound_vel_num
      end structure


      include '(fsybdb)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                  err_handler
      EXTERNAL                  msg_handler

      INTEGER*4                 login
      INTEGER*4                 dbproc
      INTEGER*4                 return_code


      INTEGER*4                 errord
      CHARACTER*(256)           cmdbuf



        record /gstation/ gstation
        record /gdata/ gdata
        record /station/ station
        record /data/ data
        record /stat/stat

        INTEGER ID_STAT,ID_DATA,STATUS,LUN
```

```
      INTEGER DIFF_DEPTH
      INTEGER LONG_MIN,LONG
      CHARACTER*30 ASCII_TIME
      CHARACTER*4 YEAR
      CHARACTER*2 DAY
      CHARACTER*2 HOUR
      CHARACTER*3 MONTH(12)
      INTEGER*2 LEAP_YEAR


      DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
     2           'Sep','Oct','Nov','Dec'/

      stat.s_num=0
      stat.o2_num=0
      stat.ipo4_num=0
      stat.tpo4_num=0
      stat.sio4_num=0
      stat.no3_num=0
      stat.no4_num=0
      stat.ph_num=0
      stat.quality_flag=0
      stat.sigma_num=0
      stat.sound_vel_num=0



C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C

      login = fdblogin()
      call fdbsetluser(login, 'REINKE')
      call fdbsetlpwd(login, 'ihlea')
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)

        call fdbuse(dbproc,'SouthernOceanDB')

        STATUS=LIB$GET_LUN(LUN)

        OPEN(LUN,FILE='OTH$DATEN:[OZEDB.GORDON]GORDON.DAT',
        1     STATUS='OLD', ACCESS='SEQUENTIAL', RECL = 80,
        1     CARRIAGECONTROL='FORTRAN')
C       1     RECL=80, FORM='FORMATTED',RECORDTYPE='FIXED')
c
C       *****Zaehlung der  Records
        ID_STAT=100000
        ID_DATA=1000000

C       *****Eroeffnen der Transaktion

100     FORMAT(15X,A11,A1,I2,I3,A1,A3,A3,X,3I2,I3,A6,I5,10X,2I4,2X)
```

```
110     FORMAT(A4,2X,A8,A47,I3,2X,I3)
120     FORMAT(I5,I1,X,2(I5,2I1),I4,I1,I5,I1,I4,2I1,10X,
        1       3(I4,I1),3(I3,I1),A3)

10      CONTINUE
C       *****Lesen des Files
        READ(LUN,100,END=3)
        1       GSTATION.SQUARE_STRING,
        1       GSTATION.LATI_NAME,
        1       GSTATION.LATI,
        1       GSTATION.LATI_MIN,
        1       GSTATION.LONG_NAME,
        1       GSTATION.LONG,
        1       GSTATION.LONG_MIN,
        1       GSTATION.YEAR,
        1       GSTATION.MONTH,
        1       GSTATION.DAY,
        1       GSTATION.HOUR,
        1       GSTATION.SHIP_NAME,
        1       GSTATION.BOTTOM_DEPTH,
        1       GSTATION.MIN_OBSE_DEPTH,
        1       GSTATION.MAX_OBSE_DEPTH

        READ(LUN,110)
        1       GSTATION.DIFF_DEPTH,
        1       GSTATION.OPTION,
        1       GSTATION.ETC_STRING,
        1       GSTATION.NUMBER_OBSE,
        1       GSTATION.NUMBER_STD_DEPTH

C       ***Behandlung von GSTATION.DIFF_DEPTH

        DIFF_DEPTH=0
        LONG=-9999
        LONG_MIN=0
        if (GSTATION.DIFF_DEPTH .ne. '****')
        1       read (GSTATION.DIFF_DEPTH,'(I4)') DIFF_DEPTH
        if (GSTATION.LONG .ne. '-40') THEN
                read (GSTATION.LONG,'(I4)') LONG
                read (GSTATION.LONG_MIN,'(I4)') LONG_MIN
        END IF

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********


        GSTATION.YEAR=GSTATION.YEAR + 1900
        LEAP_YEAR = mod(GSTATION.YEAR,4)

        if ((GSTATION.HOUR .gt.24 .or. GSTATION.HOUR .lt. 0) .OR.
        1   (GSTATION.DAY  .gt.31 .or. GSTATION.DAY  .lt. 1) .OR.
        1   (GSTATION.MONTH.gt.12 .or. GSTATION.MONTH.lt. 1) .OR.
        1   (GSTATION.YEAR .gt.1989 .or. GSTATION.YEAR.lt. 1900)) then

        GSTATION.MONTH = 1
        YEAR = '1900'
        DAY  = ' 1'
        HOUR ='00'


C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (GSTATION.DAY.eq. 29 .and.
        1        GSTATION.MONTH.eq. 2 .and.
        1        LEAP_YEAR.ne.0) THEN
```

```
          GSTATION.MONTH= 1
          YEAR = '1900'
          DAY = ' 1'
          HOUR ='00'

         ELSE



          WRITE (DAY,'(I2)') GSTATION.DAY
          WRITE (YEAR,'(I4)')GSTATION.YEAR
          IF (GSTATION.HOUR .eq. 24) THEN
           HOUR ='23'
          ELSE
           WRITE (HOUR,'(I2)') GSTATION.HOUR
          END IF
         END IF

         ASCII_TIME='"'//MONTH(GSTATION.MONTH)//' '//DAY//' '//YEAR//' '
         2//HOUR//':00'//'"'

C        ***Speicherung der Stationsdaten***************

         ID_STAT=ID_STAT+1

         STATION.ID=ID_STAT
         STATION.SHIP_NAME=GSTATION.SHIP_NAME
         STATION.LATITUDE=
         1   dfloat(GSTATION.LATI)+dfloat(GSTATION.LATI_MIN)/10./60.
         if (GSTATION.LATI_NAME .eq.'S')
         1               STATION.LATITUDE=(-1.)*STATION.LATITUDE

         STATION.LONGITUDE=
         1   dfloat(LONG)+dfloat(LONG_MIN)/10./60.
         if (GSTATION.LONG_NAME .eq.'W')
         1               STATION.LONGITUDE=(-1.)*STATION.LONGITUDE
         STATION.BOTTOM_DEPTH=GSTATION.BOTTOM_DEPTH
         STATION.MAX_OBSE_DEPTH=GSTATION.MAX_OBSE_DEPTH
         STATION.NUMBER_OBSE=GSTATION.NUMBER_OBSE
C
         type *, station.id,' ',ascii_time

c        *********Statitistik***************
c        if (gstation.option(1:1) .eq. '9') stat.s_num=stat.s_num+1
c        if (gstation.option(2:2) .eq. '9') stat.o2_num=stat.o2_num+1
c        if (gstation.option(3:3) .eq. '9')
c        1                         stat.ipo4_num=stat.ipo4_num+1
c        if (gstation.option(4:4) .eq. '9')
c        1                         stat.tpo4_num=stat.tpo4_num+1
c        if (gstation.option(5:5) .eq. '9')
c        1                         stat.sio4_num=stat.sio4_num+1
c        if (gstation.option(6:6) .eq. '9') stat.no3_num=stat.no3_num+1
c        if (gstation.option(7:7) .eq. '9') stat.no4_num=stat.no4_num+1
c        if (gstation.option(8:8) .eq. '9') stat.ph_num=stat.ph_num+1
c


         call fdbcmd(dbproc,' insert into Gordon_Station values ( ')
         call fdbfcmd(dbproc,' %d,', STATION.ID)
         call fdbfcmd(dbproc,' %d,', STATION.SHIP_NAME)
         call fdbfcmd(dbproc,' %s,', ASCII_TIME)
         call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
         call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
         call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
         call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
         call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
```

```
      call fdbfcmd(dbproc,' %d,', STATION.MARSDEN_SQUARE)
      call fdbcmd(dbproc,'0,0)')



      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)

C     ********Speicherung der Messdaten*******

      DO I=1, GSTATION.NUMBER_OBSE

      READ(LUN,120,END=3)
     1       GDATA.DEPTH,
     1       GDATA.DEPTH_QUAL,
     1       GDATA.TEMP,
     1       GDATA.TEMP_PREC,
     1       GDATA.TEMP_QUAL,
     1       GDATA.SAL,
     1       GDATA.SAL_PREC,
     1       GDATA.SAL_QUAL,
     1       GDATA.SIGMA,
     1       GDATA.SIGMA_QUAL,
     1       GDATA.SOUND_VEL,
     1       GDATA.SOUND_VEL_PREC,
     1       GDATA.OXYGEN,
     1       GDATA.OXYGEN_PREC,
     1       GDATA.OXYGEN_QUAL,
     1       GDATA.IPO4,
     1       GDATA.IPO4_PREC,
     1       GDATA.TOP4,
     1       GDATA.TPO4_PREC,
     1       GDATA.SIO4,
     1       GDATA.SIO4_PREC,
     1       GDATA.NO2,
     1       GDATA.NO2_PREC,
     1       GDATA.NO3,
     1       GDATA.NO3_PREC,
     1       GDATA.PH,
     1       GDATA.PH_PREC,
     1       GDATA.OBSE_FLAG

c     *Fehlende Werte werden zunaechst auf -9999. gesetzt, spaeter
c     *auf NULL *


      if (GDATA.TEMP_PREC.EQ.0) GDATA.TEMP=-9999
      if (GDATA.SAL_PREC.EQ.0) GDATA.SAL=-9999
      if (GDATA.OXYGEN_PREC.EQ.0) GDATA.OXYGEN=-9999
      if (GDATA.IPO4_PREC.EQ.0) GDATA.IPO4=-9999
      if (GDATA.SIO4_PREC.EQ.0) GDATA.SIO4=-9999
      if (GDATA.NO3_PREC.EQ.0) GDATA.NO3=-9999


      ID_DATA=ID_DATA+1
      DATA.ID=ID_DATA
      DATA.GORDON_STATION_ID=ID_STAT
      DATA.DEPTH=GDATA.DEPTH
      DATA.TEMPERATURE=dfloat(GDATA.TEMP)/10**(GDATA.TEMP_PREC)
      DATA.SALINITY=dfloat(GDATA.SAL)/10**(GDATA.SAL_PREC)
      DATA.OXYGEN=dfloat(GDATA.OXYGEN)/10**(GDATA.OXYGEN_PREC)
      DATA.IPO4=dfloat(GDATA.IPO4)/10**(GDATA.IPO4_PREC)
      DATA.SIO4=dfloat(GDATA.SIO4)/10**(GDATA.SIO4_PREC)
      DATA.NO3=dfloat(GDATA.NO3)/10**(GDATA.NO3_PREC)
```

```
c       ************Statistik
c       if(gdata.depth_qual.ne.0) stat.quality_flag=stat.quality_flag+1
c       if(gdata.temp_qual.ne.0) stat.quality_flag=stat.quality_flag+1
c       if(gdata.sal_qual.ne.0) stat.quality_flag=stat.quality_flag+1
c       if(gdata.sigma_qual.ne.0) stat.quality_flag=stat.quality_flag+1
c       if(gdata.oxygen_qual.ne.0) stat.quality_flag=stat.quality_flag+1
c
c       if(gdata.sound_vel.ne.0) stat.sound_vel_num=stat.sound_vel_num+1
c       if(gdata.sigma.ne.0) stat.sigma_num=stat.sigma_num+1


        call fdbcmd(dbproc,' insert into Gordon_Standard_Data values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.GORDON_STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f,', DATA.OXYGEN )
        call fdbfcmd(dbproc,' %f,', DATA.IPO4 )
        call fdbfcmd(dbproc,' %f,', DATA.SIO4 )
        call fdbfcmd(dbproc,' %f,', DATA.NO3 )
        call fdbcmd(dbproc,' 0,0)')

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)


        END DO




        GOTO 10
3       CONTINUE
        TYPE *,'end of file'
        TYPE *,' there are ', i6, ' stations in the file' ,ISTAT
        CLOSE(LUN)
        CLOSE(LUN1)
        call fdbexit()
        STATUS=LIB$GET_LUN(LUN)
        OPEN(LUN,FILE='GORDON_STAT',STATUS='NEW')
        CLOSE(LUN)
        END

C
C       ERR_HANDLER - This funtion may be coded within the same program
C       or as a separate file that is compiled/linked.
C

        INTEGER*4 FUNCTION err_handler (dbproc, severity, errno, oserrno)
C
        include '(fsybdb)'
C
        INTEGER*4       dbproc
        INTEGER*4       severity
        INTEGER*4       errno
        INTEGER*4       oserrno
        INTEGER*4       length
        INTEGER*4       return_code
C
        CHARACTER*(80)  message
C
```

```
          length = fdberrstr(errno,message)
          type *, 'DB-LIBRARY error: ', message
C
C     Check for operating system errors
C
          length = 0
          message = ' '
          length = fdboserrstr(oserrno, message)
C
          if (oserrno .ne. DBNOERR) then
             type *, 'Operating-system error: ', message
          end if
C
          return_code = fdbdead(dbproc)
C
          if ((dbproc .eq. NULL) .OR. (return_code ) .OR.
     2       (severity .eq. EXSERVER)) then
             err_handler = INT_EXIT
C
          else
             err_handler = INT_CANCEL
          end if
C
          END
C
C MSG_HANDLER - This funtion may be coded within the same program
C               or as a separate file that is compiled/linked.
C

      INTEGER*4 FUNCTION msg_handler (dbproc, msgno,
     2            msgstate,severity, msgtext)
C
      include '(fsybdb)'
C
      INTEGER*4       dbproc
      INTEGER*4       msgno
      INTEGER*4       msgstate
      INTEGER*4       severity
C
      CHARACTER*80    msgtext
        IF (MSGNO.NE.5701) THEN
C
          type *, 'DataServer message ', msgno,
     2       '   state ', msgstate, '    severity ',
     3       severity,' ', msgtext
C
        END IF
         msg_handler = DBNOSAVE

      END
```

AARILOAD.FOR
OTH$DATEN:[OZEDB.DATALOAD]
Oze db_Sybase 1.FOR

Basis für Lade programm

Ladeprog:

[~~SO46~~ OZEDB.DATA]

Aari Stat.

```
      options /check=all
      program ozedb_load

C     CREATOR::M. Reinke
C     CREA_DATE::25-Jul-1990
      structure /station/
      integer *4        ID
      integer *4        CRUISE_NUMBER
      integer *4        STATION_NUMBER
      real *8           LATITUDE
      real *8           LONGITUDE
      integer *4        BOTTOM_DEPTH
      integer *4        MAX_OBSE_DEPTH
      integer *4        NUMBER_OBSE
      integer *4        MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4         ID
      integer*4         AARI_Station_ID
      real*8            TEMPERATURE
      real*8            SALINITY
      real*8            OXYGEN
      integer*4         DEPTH
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                  err_handler
      EXTERNAL                  msg_handler

        INTEGER*4                 login,
        1                         dbproc,
        1                         return_code,
        1                         no_echo,
        1                         lun,
        1                         ipb,
        1                         id_stat,
        1                         id_data,
        1                         leap_year,
        1                         monat,
        1                         i

        character*4               Jahr
        character*2               Tag,
        1                         Stunde
        character*3               month(12)

        character*30              ASCII_TIME

      INTEGER*4                 error
      CHARACTER*(256)           cmdbuf


        CHARACTER*20 password

        INTEGER*4  nseq,
```

FOR-3

```
      1     nc,
      1     ns

      REAL*8  ongitud,
      1     atitud

      INTEGER*4  nyear,
      1          nmo,
      1          nda,
      1          nho,
      1          nde,
      1          mode,
      1          nz,
      1          msq,
      1          ni

      character file1*50

C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

      DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
      2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

C     Guretsky, AWI, 21 June 1990
C
  101 format(2x,3i7,2f8.2,9i7)
```

```
  102 format(2x,i4,x,3f8.3)

15    format(' Name of the input file: '$)
20    format(a50)
      type 15
      accept 20, file1
      call lib$get_lun(lun)
      open(unit=lun, file=file1,status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Aari_Station_Id#) from Aari_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        call fdbfcmd(dbproc,
        1  'select max(Aari_Standard_Data_Id#) from Aari_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

  222 continue
      read(lun,101,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda, nho, nde, mode, nz, msq

      read(lun,101) ni

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1989 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'

        ELSE

         WRITE (TAG,'(I2)') nda
         WRITE (JAHR,'(I4)') nyear
         IF (nho .eq. 24) THEN
          Stunde ='23'
         ELSE
          WRITE (STUNDE,'(I2)') nho
         END IF
```

```
 MONAT=nmo
END IF

ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
2//STUNDE//':00'//'"'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into Aari_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d,', STATION.MARSDEN_SQUARE)
        call fdbcmd(dbproc,'0,0)')

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)
C       ********Speicherung der Messdaten*******

        do i=1,ni
        read(lun,102) DATA.DEPTH,
        1             DATA.TEMPERATURE,
        1             DATA.SALINITY,
        1             DATA.OXYGEN

        id_data=id_data+1
        DATA.ID=id_data
        DATA.AARI_STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into Aari_Standard_Data values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.AARI_STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f,', DATA.OXYGEN )
        call fdbcmd(dbproc,' 0,0)')
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO

        GOTO 222
333     CONTINUE
        TYPE *,'end of file'
        TYPE *,' there are ',ID_STAT, ' stations in the file'
        CLOSE(LUN)
```

```
call fdbexit()
END
```

Oze_Sybase 2.FOR
10.9.1990

```
        options /check=all
        program ozedb_load

C       CREATOR::M. Reinke
C       CREA_DATE::10-sep-1990

        structure /data/
        integer*4       ID
        integer*4       GORDON_Station_ID
        real*8          TEMPERATURE
        real*8          SALINITY
        real*8          OXYGEN
        integer*4       DEPTH
        end structure

        record /DATA/ DATA


        include '(fsybdb)'
        include '($smgdef)'
        include '($ttdef)'
        include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                  err_handler
      EXTERNAL                  msg_handler

        INTEGER*4                 login,
        1                         dbproc,
        1                         return_code,
        1                         no_echo,
        1                         lun,
        1                         ipb,
        1                         i


      INTEGER*4                 error
      CHARACTER*(256)           cmdbuf


        CHARACTER*30 password


      character file1*50

c       ---declarations from oth$daten:[socean.for]read2.for


      real*8 temg(42),salg(42),oxyg(42)
      integer*4 zst(42)
C
        integer n,IDG,mmax


C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
```

→ fordas_interpol/_data.

[SOCEAN]INTERGOR.DAT

FOR-4

```
C     to open a connection to the DataServer.
C

      login = fdblogin()
      call fdbsetluser(login, 'sa')

c       **********ask for password*****************
5       FORMAT(' Password for sa: '$)
10      FORMAT (a30)

        call smg$create_pasteboard(ipb)
        no_echo=tt$m_noecho
        call smg$set_term_characteristics(ipb,no_echo)
        type 5
        accept 10,password
        call smg$set_term_characteristics(ipb,,,no_echo)

      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

C     Guretsky, AWI, 21 June 1990
C

15    format(' Name of the input file: '$)
20    format(a50)
      type 15
      accept 20, file1
      call lib$get_lun(lun)
      open(unit=lun, file=file1,status='old')

        Data.id=100000000

  222 continue

      read(lun,99,end=333) n, IDG, mmax

        do i=1,mmax
         read(lun,300,end=333) zst(i), temg(i), salg(i), oxyg(i)
        end do


   99 format(2x,i4,2x,i7,2x,i2)
  300 format(2x,i4,x,3(2x,f8.4))



C       ********Speicherung der Messdaten*******

        do i=1,mmax
        DATA.DEPTH = zst(i)
        DATA.TEMPERATURE = dble(temg(i))
        DATA.SALINITY= dble(salg(i))
        DATA.OXYGEN = dble(oxyg(i))

        DATA.GORDON_STATION_ID = IDG
        DATA.ID = Data.Id+1
```

```
        if (mod(data.gordon_Station_id,100).eq.0) then
1000    format(i10,i8,2x,i4,3(x,f10.2))
          type 1000,Data.id,data.Gordon_Station_Id,data.depth,
        1    data.temperature,data.salinity,data.oxygen
        end if

        call fdbcmd(dbproc,' insert into Gordon_Interpolated_Data values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.GORDON_STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f,', DATA.OXYGEN )
        call fdbcmd(dbproc,' 0,0)')
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)


        END DO


        GOTO 222
333     CONTINUE
        TYPE *,'end of file'
        CLOSE(LUN)
        call fdbexit()
        END

C
C     ERR_HANDLER - This funtion may be coded within the same program
C     or as a separate file that is compiled/linked.
C

      INTEGER*4 FUNCTION err_handler (dbproc, severity, errno, oserrno)
C
      include '(fsybdb)'
C
      INTEGER*4       dbproc
      INTEGER*4       severity
      INTEGER*4       errno
      INTEGER*4       oserrno
      INTEGER*4       length
      INTEGER*4       return_code
C
      CHARACTER*(80)  message
C
         length = fdberrstr(errno,message)
         type *, 'DB-LIBRARY error: ', message
C
C     Check for operating system errors
C
         length = 0
         message = ' '
         length = fdboserrstr(oserrno, message)
C
         if (oserrno .ne. DBNOERR) then
            type *, 'Operating-system error: ', message
         end if
C
         return_code = fdbdead(dbproc)
C
         if ((dbproc .eq. NULL) .OR. (return_code ) .OR.
     2      (severity .eq. EXSERVER)) then
            err_handler = INT_EXIT
C
         else
```

```
            err_handler = INT_CANCEL
         end if
C
         END
C
C MSG_HANDLER - This funtion may be coded within the same program
C               or as a separate file that is compiled/linked.
C

      INTEGER*4 FUNCTION msg_handler (dbproc, msgno,
     2            msgstate,severity, msgtext)
C
      include '(fsybdb)'
C
      INTEGER*4       dbproc
      INTEGER*4       msgno
      INTEGER*4       msgstate
      INTEGER*4       severity
C
      CHARACTER*80    msgtext
        IF (MSGNO.NE.5701) THEN
C
         type *, 'DataServer message ', msgno,
     2       '   state ', msgstate, '    severity ',
     3       severity,' ', msgtext
C
        END IF
         msg_handler = DBNOSAVE

       END
```

Mars.FOR
16.10.90

```
      SUBROUTINE MARS(ALAT,ALON,MSQ)
C
C     Calculate Marsden square number for the given
C     Latitude (ALAT)  and Longitude (Alon)
C     ONLY FOR THE SOUTHERN HEMISPHERE
C     THE NORTHERN AND THE EASTERN BOUNDARIES ARE ASSUMED
C     TO BELONG TO THE CORRESPONDING MARSDEN SQUARES
C
      A=ALAT/10.
      A=ABS(A)
      NLA=int(A)
      A=ALON/10.
      A=ABS(A)
      NLO=int(A)
      if(ALON.gt.-180..and.alon.le.0.) GO TO 1
      GO TO 2
C            *** WESTERN HEMISPHERE
    1 MSQ=36*NLA + NLO+300
      GO TO 3
C            *** EASTERN HEMISPHERE
    2 continue
      C=ABS(ALON)
      D=C/10.
      E=AINT(D)
      IF(E-D)4,5,4
C       POINT IS NOT ON THE LINE OF ROUND LONGITUDE
    4 MSQ=36*NLA + 335 -NLO
      go to 3
C       POINT IS ON THE LINE OF ROUND LONGITUDE
    5 MSQ=36*NLA + 334 - NLO
    3 continue
      return
      end
```

FOR-5

```
DEFAULT_FONT_PATH=/usr/local/tex/fonts
DEFAULT_FONT_SIZES=300:328.6:360:432:518.4:622:746.4
DEFINES=-DMSBITFIRST \
  -DBMSHORT
FONTDEFINES=-DDEFAULT_FONT_PATH=\"$(DEFAULT_FONT_PATH)\" \
  -DDEFAULT_FONT_SIZES=\"$(DEFAULT_FONT_SIZES)\"
FONTFORMATS_C=gf.c pk.c pxl.c
FONTFORMATS_O=gf.o pk.o pxl.o

DEPLIBS=XawClientDepLibs
LOCAL_LIBRARIES=XawClientLibs
MATHLIB=-lm
SYS_LIBRARIES=$(MATHLIB)
SRCS=xdvi.c dvi_init.c dvi_draw.c $(FONTFORMATS_C) pxl_open.c tpic.c
OBJS=xdvi.o dvi_init.o dvi_draw.o $(FONTFORMATS_O) pxl_open.o tpic.o

ComplexProgramTarget(xdvi)

pxl_open.o:
        $(CC) -c $(CFLAGS) $(FONTDEFINES) pxl_open.c

xdvi.man: xdvi_man.sed
        chmod u+x mksedscript
        mksedscript $(DEFAULT_FONT_PATH) $(DEFAULT_FONT_SIZES) $(DEFINES) \
                > sedscript
        sed -f sedscript < xdvi_man.sed > xdvi.man

clean::
        $(RM) sedscript xdvi.man xdvi10.man.s

lint::
        $(LINT) $(INCLUDES) $(DEFINES) $(FONTDEFINES) $(SRCS)
```

NOWLIN.FOR
Nowlin_load
8.11.90

```
      options /check=all
      program southernoceandb_Nowlin_load

C     CREATOR::M. Reinke, L.P. Kurdelski
C     CREA_DATE::25-Jul-1990
C     CHANGED::08-Nov-1990    modified to be used for Nowlin.data
C                             ERR_HANDLER include by library
C                             MSG_HANDLER include by library
C
      structure /station/
      integer *4      ID
      integer *4      CRUISE_NUMBER
      integer *4      STATION_NUMBER
      real *8         LATITUDE
      real *8         LONGITUDE
      integer *4      BOTTOM_DEPTH
      integer *4      MAX_OBSE_DEPTH
      integer *4      NUMBER_OBSE
      integer *4      MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4       ID
      integer*4       Nowlin_Station_ID
      integer*4       DEPTH
      real*8          TEMPERATURE
      real*8          SALINITY
      real*8          OXYGEN
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL              err_handler
      EXTERNAL              msg_handler

        INTEGER*4             login,
        1                     dbproc,
        1                     return_code,
        1                     no_echo,
        1                     lun,
        1                     ipb,
        1                     id_stat,
        1                     id_data,
        1                     leap_year,
        1                     monat,
        1                     i

        character*4           Jahr
        character*2           Tag,
        1                     Stunde
        character*3           month(12)


        character*30          ASCII_TIME

      INTEGER*4             error
```

```
      CHARACTER*(256)         cmdbuf

        CHARACTER*20 password

        INTEGER*4  nseq,
        1     nc,
        1     ns

        REAL*8  ongitud,
        1     atitud

        INTEGER*4  nyear,
        1          nmo,
        1          nda,
        1          nho,
        1          nde,
        1          mode,
        1          nz,
        1          msq,
        1          ni

      character file1*50

C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

        DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
        2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are linked from a library.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
C       **********ask for password*****************
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')
```

```
C       ******** reading data from disk *********
C
C     Guretsky, AWI, 21 June 1990
C
  101 format(2x,3i7,2f8.2,9i6)
  102 format(2x,i4,x,3f8.3)

15    format(' Name of the input file: '$)
20    format(a50)
      type 15
      accept 20, file1
      call lib$get_lun(lun)
      open(unit=lun, file=file1, status='old', readonly)

C       *****Zaehlung der  Records

        ID_STAT = 0
        call fdbfcmd(dbproc,
        1       'select max(Nowlin_Station_Id#) from Nowlin_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)
        if (ID_STAT .EQ. 0) then
            ID_STAT = 200000
          end if

        call fdbfcmd(dbproc,
        1  'select max(Nowlin_Standard_Data_Id#) from Nowlin_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)
        if (ID_DATA .EQ. 0) then
            ID_DATA = 20000000
          end if

  222 continue
      read(lun,101,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda, nho, nde, mode, nz, msq

      read(lun,101) ni

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1989 .or. nyear .lt. 1900)) then

            Monat = 1
            Jahr = '1900'
            Tag = ' 1'
            Stunde ='00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

            Monat = 1
            Jahr = '1900'
```

```
          Tag = ' 1'
          Stunde ='00'

        ELSE

          WRITE (TAG,'(I2)') nda
          WRITE (JAHR,'(I4)') nyear
          IF (nho .eq. 24) THEN
              Stunde ='23'
          ELSE
              WRITE (STUNDE,'(I2)') nho
          END IF
          MONAT=nmo
        END IF

        ASCII_TIME=MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':00'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into Nowlin_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' "%s",', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)
C       ********Speicherung der Messdaten*******

        do i=1,ni
        read(lun,102) DATA.DEPTH,
        1             DATA.TEMPERATURE,
        1             DATA.SALINITY,
        1             DATA.OXYGEN

        id_data=id_data+1
        DATA.ID=id_data
        DATA.Nowlin_STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into Nowlin_Standard_Data values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.Nowlin_STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f)', DATA.OXYGEN)
```

```
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO

c       ***Copy Information from Nowlin table to Station table

        type *, ' copy information Station table',ID_STAT

C       call fdbfcmd(dbproc,
C       1       'Nowlin_copy %d', ID_STAT)
C       call fdbsqlexec(dbproc)

        GOTO 222
333     CONTINUE
        TYPE *,'end of file'
        TYPE *,' there are ',ID_STAT, ' stations in the file'
        CLOSE(LUN)
        call fdbexit()
        END
```

KUROPATKIN.FOR
27.11.90

```
      options /check=all
      program ozedb_load

C     CREATOR::M. Reinke
C     CREA_DATE::25-Jul-1990
C     Loading KUROPATKIN data for modifikation of multiple
C     defined Stations and Station_Data

      structure /station/
      integer *4        ID
      integer *4        CRUISE_NUMBER
      integer *4        STATION_NUMBER
      real *8           LATITUDE
      real *8           LONGITUDE
      integer *4        BOTTOM_DEPTH
      integer *4        MAX_OBSE_DEPTH
      integer *4        NUMBER_OBSE
      integer *4        MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4         ID
      integer*4         Kuropatkin_Station_ID
      real*8            TEMPERATURE
      real*8            SALINITY
      real*8            OXYGEN
      integer*4         DEPTH
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                  err_handler
      EXTERNAL                  msg_handler

        INTEGER*4                 login,
        1                         dbproc,
        1                         return_code,
        1                         no_echo,
        1                         lun,
        1                         ipb,
        1                         id_stat,
        1                         id_data,
        1                         leap_year,
        1                         monat,
        1                         i

        character*4               Jahr
        character*2               Tag,
        1                         Stunde
        character*3               month(12)

        character*30              ASCII_TIME

      INTEGER*4                 error
      CHARACTER*(256)           cmdbuf

        CHARACTER*20 password
```

FOR-7

```
      INTEGER*4  nseq,
     1     nc,
     1     ns

      REAL*8  ongitud,
     1     atitud

      INTEGER*4  nyear,
     1           nmo,
     1           nda,
     1           nho,
     1           nde,
     1           mode,
     1           nz,
     1           msq,
     1           ni

      character file1*50

C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

      DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
     2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

C     Guretsky, AWI, 21 June 1990
```

```
C
  101 format(2x,3i7,2f8.2,9i7)
  102 format(2x,i4,x,3f8.3)

15    format(' Name of the input file: '$)
20    format(a50)
      type 15
      accept 20, file1
      call lib$get_lun(lun)
      open(unit=lun, file=file1,status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Kuropatkin_Station_Id#) from Kuropatkin_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        call fdbfcmd(dbproc,
        1  'select max(Kuropatkin_Standard_Data_Id#) ')
        call fdbfcmd(dbproc,
        1  '  from Kuropatkin_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

  222 continue
      read(lun,101,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda, nho, nde, mode, nz, msq

      read(lun,101) ni

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1989 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'

        ELSE

         WRITE (TAG,'(I2)') nda
         WRITE (JAHR,'(I4)') nyear
         IF (nho .eq. 24) THEN
```

```
          Stunde ='23'
         ELSE
          WRITE (STUNDE,'(I2)') nho
         END IF
         MONAT=nmo
        END IF

        ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':00'//'"'

C       ***Speicherung der Stationsdaten****************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into Kuropatkin_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)
C       ********Speicherung der Messdaten*******

        do i=1,ni
        read(lun,102) DATA.DEPTH,
        1             DATA.TEMPERATURE,
        1             DATA.SALINITY,
        1             DATA.OXYGEN

        id_data=id_data+1
        DATA.ID=id_data
        DATA.Kuropatkin_STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into Kuropatkin_Standard_Data values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.Kuropatkin_STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f)', DATA.OXYGEN )
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO

        GOTO 222
333     CONTINUE
        TYPE *,'end of file'
```

```
TYPE *,' there are ',ID_STAT, ' stations in the file'
CLOSE(LUN)
call fdbexit()
END
```

HAINESLOAD.FOR

28.11.90

```
      options /check=all
      program ozedb_load

C     CREATOR::M. Reinke
C     CREA_DATE::25-Jul-1990
C     CHANGES:: 1990-11-28 L.-P. Kurdelski
C                            reading Haines Lamont-Doherty data
C
      structure /station/
      integer *4       ID
      integer *4       CRUISE_NUMBER
      integer *4       STATION_NUMBER
      real *8          LATITUDE
      real *8          LONGITUDE
      integer *4       BOTTOM_DEPTH
      integer *4       MAX_OBSE_DEPTH
      integer *4       NUMBER_OBSE
      integer *4       MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4        ID
      integer*4        Haines_Station_ID
      real*8           TEMPERATURE
      real*8           SALINITY
      real*8           OXYGEN
      integer*4        DEPTH
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                 err_handler
      EXTERNAL                 msg_handler

        INTEGER*4                login,
        1                        dbproc,
        1                        return_code,
        1                        no_echo,
        1                        lun,
        1                        ipb,
        1                        id_stat,
        1                        id_data,
        1                        leap_year,
        1                        monat,
        1                        i

        character*4              Jahr
        character*2              Tag,
        1                        Stunde
        character*3              month(12)

        character*30             ASCII_TIME

      INTEGER*4                error
      CHARACTER*(256)          cmdbuf
```

FOR-8

```
      CHARACTER*20 password

      INTEGER*4  nseq,
     1     nc,
     1     ns

      REAL*8  ongitud,
     1     atitud

      INTEGER*4  nyear,
     1           nmo,
     1           nda,
     1           nho,
     1           nde,
     1           mode,
     1           nz,
     1           msq,
     1           ni

      character file1*50

C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

      DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
     2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********
```

```
C     Guretsky, AWI, 21 June 1990
C
  401 format(2x,3i7,2x,2f9.4,2x,8i5)
  102 format(2x,i4,4x,3f8.3)

15    format(' Name of the input file: '$)
20    format(a50)
      type 15
      accept 20, file1
      call lib$get_lun(lun)
      open(unit=lun, file=file1,status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Haines_Station_Id#) from Haines_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        call fdbfcmd(dbproc,
        1  'select max(Haines_Standard_Data_Id#) from Haines_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

  222 continue
      read(lun,401,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda, nho, nde, mode, nz, msq

      read(lun,401) ni

C
C       Die Haines Daten enthalten nur die Zehner- und Einerstellen
C       der Jahreszahl. Daher muss ueberprueft werden, ob diese Zahl
C       mit den einfachen Jahreszahlen vertraeglich ist.

        if (nyear .lt. 100) then
           nyear = nyear + 1900
        end if

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
```

```
 Tag = ' 1'
 Stunde ='00'

ELSE

 WRITE (TAG,'(I2)') nda
 WRITE (JAHR,'(I4)') nyear
 IF (nho .eq. 24) THEN
  Stunde ='23'
 ELSE
  WRITE (STUNDE,'(I2)') nho
 END IF
 MONAT=nmo
END IF

ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
2//STUNDE//':00'//'"'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into Haines_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)
C       ********Speicherung der Messdaten*******

        do i=1,ni
        read(lun,102) DATA.DEPTH,
        1             DATA.TEMPERATURE,
        1             DATA.SALINITY,
        1             DATA.OXYGEN

        id_data=id_data+1
        DATA.ID=id_data
        DATA.Haines_STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into Haines_Standard_Data values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.Haines_STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f)', DATA.OXYGEN )
```

```
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO

        GOTO 222
333     CONTINUE
        TYPE *,'end of file'
        TYPE *,' there are ',ID_STAT, ' stations in the file'
        CLOSE(LUN)
        call fdbexit()
        END
```

Gonella_load.FOR;10
15.12.90

```
      options /check=all
      program gonella_load

C     CREATOR::M. Reinke
C     CREA_DATE::25-Jul-1990
C     CHANGES:: 1990-12-15 L.-P. Kurdelski
C                          reading Gonella (Marion Dufresne) data
C                          National Museum of Natural History
C
C     Station_Id              400000
C     Standard_Data_Id       4000000
C
      structure /station/
      integer *4      ID
      integer *4      CRUISE_NUMBER
      integer *4      STATION_NUMBER
      real *8         LATITUDE
      real *8         LONGITUDE
      integer *4      BOTTOM_DEPTH
      integer *4      MAX_OBSE_DEPTH
      integer *4      NUMBER_OBSE
      integer *4      MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4       ID
      integer*4       Gonella_Station_ID
      real*8          TEMPERATURE
      real*8          SALINITY
      real*8          OXYGEN
      integer*4       DEPTH
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL              err_handler
      EXTERNAL              msg_handler

        INTEGER*4             login,
        1                     dbproc,
        1                     return_code,
        1                     no_echo,
        1                     lun,
        1                     ipb,
        1                     id_stat,
        1                     id_data,
        1                     leap_year,
        1                     monat,
        1                     i

        character*4           Jahr
        character*2           Tag,
        1                     Stunde
        character*3           month(12)

        character*30          ASCII_TIME
```

FOR-9

```
      INTEGER*4               error
      CHARACTER*(256)         cmdbuf


        CHARACTER*20 password

        INTEGER*4  nseq,
        1     nc,
        1     ns,
        1     j

        REAL*8  ongitud,
        1     atitud

        INTEGER*4  nyear,
        1          nmo,
        1          nda,
        1          nho,
        1          nde,
        1          mode,
        1          nz,
        1          msq,
        1          ni

      character file1*50

C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square  not availabe;e
C     ni - number of standard (interpolated) levels
C

        DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
        2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
```

```
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

C     Guretsky, AWI, 21 June 1990
C
C  format to read station data
  401 format(2x,i6,1x,i4,1x,f9.4,1x,f9.4,1x,i4,1x,i4,1x,i4,1x,
     *i2,1x,i2,1x,i2,1x,i3)
C  format to read measured data
  102 format(2x,i3,2x,i4,2x,f7.3,f7.34,f6.2)
C
C MARSDEN_SQUARE MISSING THEREFORE
C
       msq = -9999
C
C
15    format(' Name of the input file: '$)
20    format(a50)
      type 15
      accept 20, file1
      call lib$get_lun(lun)
      open(unit=lun, file=file1,status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Gonella_Station_Id#) from Gonella_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = 400000
        end if

        call fdbfcmd(dbproc,
        1  'select max(Gonella_Standard_Data_Id#) from Gonella_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = 4000000
        end if

  222 continue
C    NO MARSDEN SQUARE
C    variable list differs from allother in the other load programs
C
       read(lun,401,end=333) nc, ns, ongitud, atitud, nde, mode,
     * nyear, nmo, nda, nho, nz

       read(lun,401) ni

C
C       Die Gonella Daten enthalten nur die Zehner- und Einerstellen
C       der Jahreszahl. Daher muss ueberprueft werden, ob diese Zahl
C       mit den einfachen Jahreszahlen vertraeglich ist.

        if (nyear .lt. 100) then
           nyear = nyear + 1900
        end if
```

```
C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'

        ELSE

         WRITE (TAG,'(I2)') nda
         WRITE (JAHR,'(I4)') nyear
         IF (nho .eq. 24) THEN
          Stunde ='23'
         ELSE
          WRITE (STUNDE,'(I2)') nho
         END IF
         MONAT=nmo
        END IF

        ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':00'//'"'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into Gonella_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
```

```
      call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
      call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)
C     ********Speicherung der Messdaten*******

      do i=1,ni
      read(lun,102) j,
      1             DATA.DEPTH,
      1             DATA.TEMPERATURE,
      1             DATA.SALINITY,
      1             DATA.OXYGEN

      id_data=id_data+1
      DATA.ID=id_data
      DATA.Gonella_STATION_ID = STATION.ID

      call fdbcmd(dbproc,' insert into Gonella_Standard_Data values (')
      call fdbfcmd(dbproc,' %d,', DATA.ID)
      call fdbfcmd(dbproc,' %d,', DATA.Gonella_STATION_ID)
      call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
      call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
      call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
      call fdbfcmd(dbproc,' %f)', DATA.OXYGEN )
      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)

      END DO

      GOTO 222
333   CONTINUE
      TYPE *,'end of file'
      TYPE *,' there are ',ID_STAT, ' stations in the file'
      CLOSE(LUN)
      call fdbexit()
      END
```

Marsprob.FOR
18.12.90

```
      program Marsprob
C
C
      EXTERNAL err_handler
      External msg_handler
      include '(fsybdb) '
C
      Integer*4  dbproc, login,return_code,error,MSQ
C
      Integer*2 msdb(500),mssub(500)
C
      REAL*8 Alat8,alon8
C
      REAL*4 A(500), B(500)
C
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
      login=fdblogin()
      call fdbsetluser(login,'SOCEAN')
      call fdbsetlpwd(login,'Victor')
      dbproc=fdbopen(login,NULL)
      call fdbuse(dbproc,'SouthernOceanDB')
C++++++++=============================================
C
      call fdbfcmd(dbproc,'Execute Marsprob')
      call fdbsqlexec(dbproc)
      call fdbresults(dbproc)
      call fdbsetnull(dbproc,intbind,0,0)
      call fdbbind(dbproc,1,flt8bind,0,Alat8)
      call fdbbind(dbproc,2,flt8bind,0,Alon8)
      call fdbbind(dbproc,3,intbind,0,MSQ)
      II=0
      do while(fdbnextrow(dbproc).ne.NO_MORE_ROWS)
      II=II+1
      MSDB(II)= MSQ
      A(II)=sngl(Alat8)
      B(II)=sngl(Alon8)
      call mars(a(ii),b(ii),MSSUB(II))
      end do
      call fdbexit()
      do 5 i=1,II
      type 333,i,a(i),B(i),MSDB(i),MSSUB(i)
    5 continue
  333 format(2x,i3,2f8.2,2i6)
      stop ' E N D '
      end
C ---------------------------------------

C     Error und Message Handler fuer
C     embedded SQL-Programme. In diesen mit
C     INCLUDE '(ERRMSG)' includen.
C
C     Error Handler
C     -------------
C     ERR_HANDLER - This funtion may be coded within the same program
C     or as a separate file that is compiled/linked.
C
      INTEGER*4 FUNCTION err_handler (dbproc, severity, errno, oserrno)
C
       include '(fsybdb)'
C
C      EXTERNAL        err_handler
C      EXTERNAL        msg_handler
C
      INTEGER*4       dbproc
```

FOR-10

```
      INTEGER*4       severity
      INTEGER*4       errno
      INTEGER*4       oserrno
      INTEGER*4       length
      INTEGER*4       return_code
C
      CHARACTER*(80)  message
C
         length = fdberrstr(errno,message)
         type *, 'DB-LIBRARY error: ', message
C
C     Check for operating system errors
C
         length = 0
         message = ' '
         length = fdboserrstr(oserrno, message)
C
         if (oserrno .ne. DBNOERR) then
            type *, 'Operating-system error: ', message
         end if
C
         return_code = fdbdead(dbproc)
C
         if ((dbproc .eq. NULL) .OR. (return_code ) .OR.
     2      (severity .eq. EXSERVER)) then
            err_handler = INT_EXIT
C
         else
            err_handler = INT_CANCEL
         end if
C
         END
C
C      Message Handler
C      ---------------
C MSG_HANDLER - This funtion may be coded within the same program
C               or as a separate file that is compiled/linked.
C
      INTEGER*4 FUNCTION msg_handler (dbproc, msgno,
     2            msgstate,severity, msgtext)
C
      include '(fsybdb)'
C
      INTEGER*4       dbproc
      INTEGER*4       msgno
      INTEGER*4       msgstate
      INTEGER*4       severity
C
      CHARACTER*80    msgtext
        IF (MSGNO.NE.5701) THEN
C
         type *, 'DataServer message ', msgno,
     2      '   state ', msgstate, '    severity ',
     3      severity,' ', msgtext
C
        END IF
         msg_handler = DBNOSAVE
C
      end
```

A1111 (load.com) (25.7.90) i4
13.2.91

```
        options /check=all

C       CREATOR::M. REIKNE
C       CREA_DATE::25-Jul-1990
C       CHANGES:: 1991-02-13 L.-P. Kurdelski
C                                reading new Aari data form A1111.dat
C
        structure /station/
        integer *4      ID
        integer *4      CRUISE_NUMBER
        integer *4      STATION_NUMBER
        real *8         LATITUDE
        real *8         LONGITUDE
        integer *4      BOTTOM_DEPTH
        integer *4      MAX_OBSE_DEPTH
        integer *4      NUMBER_OBSE
        integer *4      MARSDEN_SQUARE
        end structure

        structure /data/
        integer*4       ID
        integer*4       Station_ID
        real*8          TEMPERATURE
        real*8          SALINITY
        real*8          OXYGEN
        integer*4       DEPTH
        end structure

        record /STATION/ STATION
        record /DATA/ DATA

        include '(fsybdb)'
        include '($smgdef)'
        include '($ttdef)'
        include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                err_handler
      EXTERNAL                msg_handler

        INTEGER*4               login,
        1                       dbproc,
        1                       return_code,
        1                       no_echo,
        1                       lun,
        1                       ipb,
        1                       id_stat,
        1                       id_data,
        1                       leap_year,
        1                       monat,
        1                       i

        character*4             Jahr
        character*2             Tag,
        1                       Stunde,
        1                       Minute
        character*3             month(12)

        character*30            ASCII_TIME

      INTEGER*4               error
      CHARACTER*(256)         cmdbuf
```

FOR-11

```
      CHARACTER*20 password

      INTEGER*4  nseq,
     1     nc,
     1     ns

      REAL*8  ongitud,
     1     atitud

      INTEGER*4  nyear,
     1           nmo,
     1           nda,
     1           nho,
     1           nmin,
     1           nde,
     1           mode,
     1           nz,
     1           msq,
     1           ni

      character file1*50

C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nmin - minute
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

      DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
     2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')
```

```
c       ******** reading data from disk *********

C     Guretsky, AWI, 21 June 1990
C
  401 format(2x,3i7,2f8.2,9i7)
  102 format(2x,i4,1x,3f8.3)

15    format(' Name of the input file: '$)
20    format(a50)
      type 15
      accept 20, file1
      call lib$get_lun(lun)
      open(unit=lun, file=file1,status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Station_Id#) from A1111_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = 600000
        end if

        call fdbfcmd(dbproc,
        1  'select max(Standard_Data_Id#) from A1111_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = 6000000
        end if

  222 continue
      read(lun,401,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda, nho, nde, mode, nz, msq

      read(lun,401) ni

C
C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if (((nho.gt.24 .and. nho.ne.99) .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute = '00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN
```

```
 Monat = 1
 Jahr = '1900'
 Tag = ' 1'
 Stunde ='00'
ELSE

 WRITE (TAG,'(I2)') nda
 WRITE (JAHR,'(I4)') nyear
 IF (nho .eq. 24) THEN
  Stunde ='23'
 ELSE
  IF (nho .eq. 99) THEN
      STUNDE = '00'
    ELSE
      WRITE (STUNDE,'(I2)') nho
  END IF
 END IF
 MONAT=nmo
END IF

ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
2//STUNDE//':00"'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into A1111_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)
C       ********Speicherung der Messdaten*******

        do i=1,ni
        read(lun,102) DATA.DEPTH,
        1             DATA.TEMPERATURE,
        1             DATA.SALINITY,
        1             DATA.OXYGEN

        id_data=id_data+1
        DATA.ID=id_data
        DATA.STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into A1111_Standard_Data')
```

```
        call fdbfcmd(dbproc,' values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f)', DATA.OXYGEN )
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO

        GOTO 222
333     CONTINUE
        TYPE *,'end of file'
        TYPE *,' there are ',ID_STAT, ' stations in the file'
        CLOSE(LUN)
        call fdbexit()
        END
```

Tokyo load FOR.11
13.2.91

```
      options /check=all

C     CREATOR::M. REIKNE
C     CREA_DATE::25-Jul-1990
C     CHANGES:: 1991-02-13 L.-P. Kurdelski
C                          reading Tkyo Fisheries data
C
      structure /station/
      integer *4       ID
      integer *4       CRUISE_NUMBER
      integer *4       STATION_NUMBER
      real *8          LATITUDE
      real *8          LONGITUDE
      integer *4       BOTTOM_DEPTH
      integer *4       MAX_OBSE_DEPTH
      integer *4       NUMBER_OBSE
      integer *4       MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4        ID
      integer*4        Station_ID
      real*8           TEMPERATURE
      real*8           SALINITY
      real*8           OXYGEN
      integer*4        DEPTH
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                 err_handler
      EXTERNAL                 msg_handler

        INTEGER*4                login,
        1                        dbproc,
        1                        return_code,
        1                        no_echo,
        1                        lun,
        1                        ipb,
        1                        id_stat,
        1                        id_data,
        1                        leap_year,
        1                        monat,
        1                        i

        character*4              Jahr
        character*2              Tag,
        1                        Stunde,
        1                        Minute
        character*3              month(12)

        character*30             ASCII_TIME

      INTEGER*4                error
      CHARACTER*(256)          cmdbuf
```

FOR-12

```
      CHARACTER*20 password

      INTEGER*4  nseq,
     1     nc,
     1     ns

      REAL*8  ongitud,
     1     atitud

      INTEGER*4  nyear,
     1           nmo,
     1           nda,
     1           nho,
     1           nmin,
     1           nde,
     1           mode,
     1           nz,
     1           msq,
     1           ni

      character file1*50

C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nmin - minute
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

      DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
     2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')
```

```
c      ******** reading data from disk *********

C      Guretsky, AWI, 21 June 1990
C
  401 format(2x,3i7,2f8.2,9i7)
  102 format(2x,i4,1x,3f8.3)

15     format(' Name of the input file: '$)
20     format(a50)
       type 15
       accept 20, file1
       call lib$get_lun(lun)
       open(unit=lun, file=file1,status='old')

C        *****Zaehlung der  Records

         call fdbfcmd(dbproc,
         1       'select max(Station_Id#) from Tokyo_Fisheries_Station')
         call fdbsqlexec(dbproc)
         call fdbresults(dbproc)
         call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
         call fdbnextrow(dbproc)

         if (ID_STAT .eq. 0) then
             ID_STAT = 500000
         end if

         call fdbfcmd(dbproc,
         1  'select max(Standard_Data_Id#) from Tokyo_Fisheries_Standard_Data')
         call fdbsqlexec(dbproc)
         call fdbresults(dbproc)
         call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
         call fdbnextrow(dbproc)

         if (ID_DATA .eq. 0) then
             ID_DATA = 5000000
         end if

  222 continue
      read(lun,401,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda, nho, nmin, nde, mode, nz, msq

      read(lun,401) ni

C
C        **Konstruktion des Zeitstrings
C        ***Testen ob Ausreisser in den Zeiten gibt ***********

         leap_year = mod(nyear,4)

         if (((nho.gt.24 .and. nho.ne.99) .or. nho .lt. 00) .OR.
         1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
         1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
         1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'
         Minute = '00'

C        ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

         ELSE IF (nda.eq.29 .and.
         1        nmo.eq. 2 .and.
         1        leap_year.ne.0) THEN
```

```
       Monat = 1
       Jahr = '1900'
       Tag = ' 1'
       Stunde ='00'
       Minute ='00'
      ELSE

       WRITE (TAG,'(I2)') nda
       WRITE (JAHR,'(I4)') nyear
       IF (nho .eq. 24) THEN
        Stunde ='23'
       ELSE
        IF (nho .eq. 99) THEN
            STUNDE = '00'
          ELSE
            WRITE (STUNDE,'(I2)') nho
        END IF
       END IF
       IF (nmin .eq. 99) THEN
           Minute = '00'
         ELSE
           WRITE (Minute,'(I2)') nmin
       END IF
       MONAT=nmo
      END IF

      ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
      2//STUNDE//':'//Minute//'"'

C     ***Speicherung der Stationsdaten****************

      ID_STAT=ID_STAT+1
      STATION.ID=ID_STAT
      STATION.CRUISE_NUMBER=nc
      STATION.STATION_NUMBER=ns
      STATION.LATITUDE=atitud
      STATION.LONGITUDE=ongitud
      STATION.BOTTOM_DEPTH=nde
      STATION.MAX_OBSE_DEPTH=mode
      STATION.NUMBER_OBSE=nz
      STATION.MARSDEN_SQUARE=msq

      type *, station.id,' ',ascii_time

      call fdbcmd(dbproc,' insert into Tokyo_Fisheries_Station values ( ')
      call fdbfcmd(dbproc,' %d,', STATION.ID)
      call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
      call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
      call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
      call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
      call fdbfcmd(dbproc,' %s,', ASCII_TIME)
      call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
      call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)
C     ********Speicherung der Messdaten*******

      do i=1,ni
      read(lun,102) DATA.DEPTH,
     1            DATA.TEMPERATURE,
     1            DATA.SALINITY,
     1            DATA.OXYGEN
```

```
      id_data=id_data+1
      DATA.ID=id_data
      DATA.STATION_ID = STATION.ID

      call fdbcmd(dbproc,' insert into Tokyo_Fisheries_Standard_Data')
      call fdbfcmd(dbproc,' values (')
      call fdbfcmd(dbproc,' %d,', DATA.ID)
      call fdbfcmd(dbproc,' %d,', DATA.STATION_ID)
      call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
      call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
      call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
      call fdbfcmd(dbproc,' %f)', DATA.OXYGEN )
      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)

      END DO

      GOTO 222
333   CONTINUE
      TYPE *,'end of file'
      TYPE *,' there are ',ID_STAT, ' stations in the file'
      CLOSE(LUN)
      call fdbexit()
      END
```

JARELOAD.FOR;9
5.21.5.91

```
      options /check=all
      program jareload

C     CREATOR::M. Reinke
C     CREA_DATE::25-Jul-1990
C     CHANGES:: 1991-05-21 L.-P. Kurdelski
C                          reading JARE data
C
      structure /station/
      integer *4       ID
      integer *4       CRUISE_NUMBER
      integer *4       STATION_NUMBER
      real *8          LATITUDE
      real *8          LONGITUDE
      integer *4       BOTTOM_DEPTH
      integer *4       MAX_OBSE_DEPTH
      integer *4       NUMBER_OBSE
      integer *4       MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4        ID
      integer*4        Jare_Station_ID
      real*8           TEMPERATURE
      real*8           SALINITY
      real*8           OXYGEN
      integer*4        DEPTH
      real*8           PHOSPHATE
      real*8           NITRATE
      real*8           SILICATE
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
CC    Forward declarations of the error-handler and message-handler
C
      EXTERNAL                 err_handler
      EXTERNAL                 msg_handler

        INTEGER*4                login,
        1                        dbproc,
        1                        return_code,
        1                        no_echo,
        1                        lun,
        1                        ipb,
        1                        id_stat,
        1                        id_data,
        1                        leap_year,
        1                        monat,
        1                        i

        character*4              Jahr
        character*2              Tag,
        1                        Stunde,
        1                        Minute
        character*3              month(12)

        character*30             ASCII_TIME
```

FOR-13

```
      INTEGER*4               error
      CHARACTER*(256)         cmdbuf


        CHARACTER*20 password

        INTEGER*4  nseq,
        1      nc,
        1      ns

        REAL*8  ongitud,
        1      atitud,
        1      phosphate,
        1      nitarte,
        1      silicate

        INTEGER*4  nyear,
        1          nmo,
        1          nda,
        1          nho,
        1          nde,
        1          mode,
        1          nz,
        1          msq,
        1          ni,
        1          nmin

      character file1*50

C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

        DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
        2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C
```

```
C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

C     Guretsky, AWI, 21 June 1990
C
  401 format(2x,3i7,2f8.2,9i7)
  102 format(2x,i4,1x,6f8.3)

15    format(' Name of the input file: '$)
20    format(a50)
      type 15
      accept 20, file1
      call lib$get_lun(lun)
      open(unit=lun, file=file1,status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Jare_Station_Id#) from Jare_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = 700000
        end if

        call fdbfcmd(dbproc,
        1  'select max(Jare_Standard_Data_Id#) from Jare_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = 7000000
        end if

  222 continue
      read(lun,401,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda, nho, nmin, nde, mode, nz, msq

      read(lun,401) ni

C
C       Die Jare Daten enthalten nur die Zehner- und Einerstellen
C       der Jahreszahl. Daher muss ueberprueft werden, ob diese Zahl
C       mit den einfachen Jahreszahlen vertraeglich ist.

        if (nyear .lt. 100) then
           nyear = nyear + 1900
        end if

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt **********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
```

```
     1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
     1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

     Monat = 1
     Jahr = '1900'
     Tag = ' 1'
     Stunde ='00'
     Minute = '00'

C    ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

     ELSE IF (nda.eq.29 .and.
     1        nmo.eq. 2 .and.
     1        leap_year.ne.0) THEN

      Monat = 1
      Jahr = '1900'
      Tag = ' 1'
      Stunde ='00'
      Minute = '00'

     ELSE

      WRITE (TAG,'(I2)') nda
      WRITE (JAHR,'(I4)') nyear
      IF (nho .eq. 24) THEN
        Stunde ='23'
       ELSE
        IF (nho .gt. 24) THEN
          STUNDE = '00'
         ELSE
          WRITE (STUNDE,'(I2)') nho
        END IF
      ENDIF
      IF (nmin .gt. 59) THEN
        Minute = '00'
       ELSE
        WRITE (Minute,'(I2)') nmin
      ENDIF
      MONAT=nmo
     END IF

     ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
     2//STUNDE//':'//Minute//'"'

C    ***Speicherung der Stationsdaten***************

     ID_STAT=ID_STAT+1
     STATION.ID=ID_STAT
     STATION.CRUISE_NUMBER=nc
     STATION.STATION_NUMBER=ns
     STATION.LATITUDE=atitud
     STATION.LONGITUDE=ongitud
     STATION.BOTTOM_DEPTH=nde
     STATION.MAX_OBSE_DEPTH=mode
     STATION.NUMBER_OBSE=nz
     STATION.MARSDEN_SQUARE=msq

     type *, station.id,' ',ascii_time

     call fdbcmd(dbproc,' insert into Jare_Station values ( ')
     call fdbfcmd(dbproc,' %d,', STATION.ID)
     call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
     call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
     call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
     call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
```

```
      call fdbfcmd(dbproc,' %s,', ASCII_TIME)
      call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
      call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)
C     ********Speicherung der Messdaten*******

      do i=1,ni
      read(lun,102) DATA.DEPTH,
     1              DATA.TEMPERATURE,
     1              DATA.SALINITY,
     1              DATA.OXYGEN,
     1              DATA.PHOSPHATE,
     1              DATA.NITRATE,
     1              DATA.SILICATE

      id_data=id_data+1
      DATA.ID=id_data
      DATA.Jare_STATION_ID = STATION.ID

      call fdbcmd(dbproc,' insert into Jare_Standard_Data values (')
      call fdbfcmd(dbproc,' %d,', DATA.ID)
      call fdbfcmd(dbproc,' %d,', DATA.Jare_STATION_ID)
      call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
      call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
      call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
      call fdbfcmd(dbproc,' %f,', DATA.OXYGEN )
      call fdbfcmd(dbproc,' %f,', DATA.PHOSPHATE)
      call fdbfcmd(dbproc,' %f,', DATA.NITRATE)
      call fdbfcmd(dbproc,' %f)', DATA.SILICATE)
      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)

      END DO

      GOTO 222
333   CONTINUE
      TYPE *,'end of file'
      TYPE *,' there are ',ID_STAT, ' stations in the file'
      CLOSE(LUN)
      call fdbexit()
      END
```

Muenchload.FOR
9.8.91

```
      options /check=all
      program Muenchload

C     CREATOR::M. Reinke
C     CREA_DATE::25-Jul-1990
C     CHANGES:: 1991-08-09 L.-P. Kurdelski
C                          reading Muench data
C
      structure /station/
      integer *4      ID
      integer *4      CRUISE_NUMBER
      integer *4      STATION_NUMBER
      real *8         LATITUDE
      real *8         LONGITUDE
      integer *4      BOTTOM_DEPTH
      integer *4      MAX_OBSE_DEPTH
      integer *4      NUMBER_OBSE
      integer *4      MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4       ID
      integer*4       Muench_Station_ID
      real*8          TEMPERATURE
      real*8          SALINITY
      integer*4       DEPTH
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL              err_handler
      EXTERNAL              msg_handler

        INTEGER*4             login,
        1                     dbproc,
        1                     return_code,
        1                     no_echo,
        1                     lun,
        1                     ipb,
        1                     id_stat,
        1                     id_data,
        1                     leap_year,
        1                     monat,
        1                     i

        character*4           Jahr
        character*2           Tag,
        1                     Stunde,
        1                     Minute
        character*3           month(12)

        character*30          ASCII_TIME

      INTEGER*4             error
      CHARACTER*(256)       cmdbuf

        CHARACTER*20 password
```

FOR-14

```
      REAL*8  ongitud,
     1     atitud

      INTEGER*4  nyear,
     1           nmo,
     1           nda,
     1           nho,
     1           nde,
     1           mode,
     1           nz,
     1           msq,
     1           ni,
     1           nmin,
     1           nseq,
     1           nc,
     1           ns

      character file1*50
C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nmin - minute
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

      DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
     2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

  400 format(2x,3i7,2f8.2)
  401 format(9i7)
  402 format(2x,i3)
  102 format(2x,i3,2x,i4,2x,2f7.3)
```

```
 15     format(' Name of the input file: '$)
 20     format(a50)
        type 15
        accept 20, file1
        call lib$get_lun(lun)
        open(unit=lun, file=file1,status='old')

 C        *****Zaehlung der  Records

          call fdbfcmd(dbproc,
          1       'select max(Muench_Station_Id#) from Muench_Station')
          call fdbsqlexec(dbproc)
          call fdbresults(dbproc)
          call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
          call fdbnextrow(dbproc)

          if (ID_STAT .eq. 0) then
              ID_STAT = 900000
          end if

          call fdbfcmd(dbproc,
          1  'select max(Muench_Standard_Data_Id#) from Muench_Standard_Data')
          call fdbsqlexec(dbproc)
          call fdbresults(dbproc)
          call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
          call fdbnextrow(dbproc)

          if (ID_DATA .eq. 0) then
              ID_DATA = 9000000
          end if

   222  continue
        read(lun,400,end=333) nseq, nc, ns, ongitud, atitud
        read(lun,401,end=333) nyear, nmo, nda, nho, nmin,
       * nde, mode, nz, msq

        read(lun,402,end=333) ni
 C
 C        Die Muench Daten enthalten nur die Zehner- und Einerstellen
 C        der Jahreszahl. Daher muss ueberprueft werden, ob diese Zahl
 C        mit den einfachen Jahreszahlen vertraeglich ist.

          if (nyear .lt. 100) then
             nyear = nyear + 1900
          end if

 C        **Konstruktion des Zeitstrings
 C        ***Testen ob Ausreisser in den Zeiten gibt ***********

          leap_year = mod(nyear,4)

          if ((nho.gt.24 .or. nho .lt. 00) .OR.
          1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
          1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
          1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

          Monat = 1
          Jahr = '1900'
          Tag = ' 1'
          Stunde ='00'
          Minute = '00'

 C        ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

          ELSE IF (nda.eq.29 .and.
```

```
     1        nmo.eq. 2 .and.
     1        leap_year.ne.0) THEN

      Monat = 1
      Jahr = '1900'
      Tag = ' 1'
      Stunde ='00'
      Minute = '00'

     ELSE

      WRITE (TAG,'(I2)') nda
      WRITE (JAHR,'(I4)') nyear
      IF (nho .eq. 24) THEN
        Stunde ='23'
       ELSE
        IF (nho .gt. 24) THEN
          STUNDE = '00'
         ELSE
          WRITE (STUNDE,'(I2)') nho
        END IF
      ENDIF
      IF (nmin .gt. 59) THEN
        Minute = '00'
       ELSE
        WRITE (Minute,'(I2)') nmin
      ENDIF
      MONAT=nmo
     END IF

     ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
     2//STUNDE//':'//Minute//'"'

C    ***Speicherung der Stationsdaten***************

     ID_STAT=ID_STAT+1
     STATION.ID=ID_STAT
     STATION.CRUISE_NUMBER=nc
     STATION.STATION_NUMBER=ns
     STATION.LATITUDE=atitud
     STATION.LONGITUDE=ongitud
     STATION.BOTTOM_DEPTH=nde
     STATION.MAX_OBSE_DEPTH=mode
     STATION.NUMBER_OBSE=nz
     STATION.MARSDEN_SQUARE=msq

     type *, station.id,' ',ascii_time

     call fdbcmd(dbproc,' insert into Muench_Station values ( ')
     call fdbfcmd(dbproc,' %d,', STATION.ID)
     call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
     call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
     call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
     call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
     call fdbfcmd(dbproc,' %s,', ASCII_TIME)
     call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
     call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
     call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
     call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

     call fdbsqlexec(dbproc)
     return_code = fdbresults(dbproc)
C    ********Speicherung der Messdaten*******

     do i=1,ni
     read(lun,102) lfd,
```

```
1             DATA.DEPTH,
1             DATA.TEMPERATURE,
1             DATA.SALINITY

id_data=id_data+1
DATA.ID=id_data
DATA.Muench_STATION_ID = STATION.ID

call fdbcmd(dbproc,' insert into Muench_Standard_Data values (')
call fdbfcmd(dbproc,' %d,', DATA.ID)
call fdbfcmd(dbproc,' %d,', DATA.Muench_STATION_ID)
call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
call fdbfcmd(dbproc,' %f)', DATA.SALINITY)
call fdbsqlexec(dbproc)
return_code = fdbresults(dbproc)

END DO

GOTO 222
333   CONTINUE
TYPE *,'end of file'
TYPE *,' there are ',ID_STAT, ' stations in the file'
CLOSE(LUN)
call fdbexit()
END
```

```
      options /check=all
      program Argentineload

C     CREATOR::M. Reinke
C     CREA_DATE::25-Jul-1990
C     CHANGES:: 1991-10-08 L.-P. Kurdelski
C                          reading Argentine data
C
      structure /station/
      integer *4      ID
      integer *4      CRUISE_NUMBER
      integer *4      STATION_NUMBER
      real *8         LATITUDE
      real *8         LONGITUDE
      integer *4      BOTTOM_DEPTH
      integer *4      MAX_OBSE_DEPTH
      integer *4      NUMBER_OBSE
      integer *4      MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4       ID
      integer*4       Argentine_Station_ID
      real*8          TEMPERATURE
      real*8          SALINITY
      real*8          OXYGEN
      integer*4       DEPTH
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL              err_handler
      EXTERNAL              msg_handler

        INTEGER*4             login,
        1                     dbproc,
        1                     return_code,
        1                     no_echo,
        1                     lun,
        1                     ipb,
        1                     id_stat,
        1                     id_data,
        1                     leap_year,
        1                     monat,
        1                     i

        character*4           Jahr
        character*2           Tag,
        1                     Stunde,
        1                     Minute
        character*3           month(12)

        character*30          ASCII_TIME

      INTEGER*4             error
      CHARACTER*(256)       cmdbuf
```

```
        CHARACTER*20 password

        REAL*8  ongitud,
        1     atitud

        INTEGER*4  nyear,
        1          nmo,
        1          nda,
        1          nho,
        1          nde,
        1          mode,
        1          nz,
        1          msq,
        1          ni,
        1          nmin,
        1          nseq,
        1          nc,
        1          ns

      character file1*50
C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nmin - minute
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

        DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
        2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

c       Reading nseq, nc, ns, lon, lat
  400 format(2x,3i7,2f8.2)
c       Reading ny, nm, nd, nh, nm, nde, mode, nobs, msq
```

```
  401 format(2x,9i7)
c       Reading nmax
  402 format(2x,i3)
c       Reading depth, temperature, salinity, oxygen
  102 format(2x,i4,1x,3f8.3)

      call lib$get_lun(lun)
      open(unit=lun,
     * file='oth$daten:[socean.argent]interarg2.dat',
     * status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Argentine_Station_Id#) from Argentine_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = 3000000
        end if

        call fdbfcmd(dbproc,
        1  'select max(Argentine_Standard_Data_Id#)')
        call fdbfcmd(dbproc,
        1  ' from Argentine_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = 30000000
        end if

  222 continue
      read(lun,400,end=333) nseq, nc, ns, ongitud, atitud
      read(lun,401,end=333) nyear, nmo, nda, nho, nmin,
     * nde, mode, nz, msq

      read(lun,402,end=333) ni
C
C       Die Argentine Daten enthalten komplette Jahreszahlen.
C

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute = '00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
```

```
1         nmo.eq. 2 .and.
1         leap_year.ne.0) THEN

 Monat = 1
 Jahr = '1900'
 Tag = ' 1'
 Stunde ='00'
 Minute = '00'

ELSE

 WRITE (TAG,'(I2)') nda
 WRITE (JAHR,'(I4)') nyear
 IF (nho .eq. 24) THEN
   Stunde ='23'
  ELSE
   IF (nho .gt. 24) THEN
     STUNDE = '00'
    ELSE
     WRITE (STUNDE,'(I2)') nho
   END IF
 ENDIF
 IF (nmin .gt. 59) THEN
   Minute = '00'
  ELSE
   WRITE (Minute,'(I2)') nmin
 ENDIF
 MONAT=nmo
END IF

ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
2//STUNDE//':'//Minute//'"'

C       ***Speicherung der Stationsdaten****************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into Argentine_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)
C       ********Speicherung der Messdaten*******

        do i=1,ni
        read(lun,102) DATA.DEPTH,
```

```
     1            DATA.TEMPERATURE,
     1            DATA.SALINITY,
     1            DATA.OXYGEN

     id_data=id_data+1
     DATA.ID=id_data
     DATA.Argentine_STATION_ID = STATION.ID

     call fdbcmd(dbproc,' insert into Argentine_Standard_Data values (')
     call fdbfcmd(dbproc,' %d,', DATA.ID)
     call fdbfcmd(dbproc,' %d,', DATA.Argentine_STATION_ID)
     call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
     call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
     call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
     call fdbfcmd(dbproc,' %f)', DATA.OXYGEN)
     call fdbsqlexec(dbproc)
     return_code = fdbresults(dbproc)

     END DO

     GOTO 222
333  CONTINUE
     TYPE *,'end of file'
     TYPE *,' there are ',ID_STAT, ' stations in the file'
     CLOSE(LUN)
     call fdbexit()
     END
```

FOR-16

```
      options /check=all
      program Schlitzerload

C     CREATOR::M. Reinke
C     CREA_DATE::25-Jul-1990
C     CHANGES:: 1991-12-20 L.-P. Kurdelski
C                          reading Schlitzer data
C               1991-11-06 L.-P. Kurdelski
C                          changing the datafile name
C

      parameter (statStrt = 4000000)
      parameter (datStrt  = 40000000)
      structure /station/
      integer *4       ID
      integer *4       CRUISE_NUMBER
      integer *4       STATION_NUMBER
      real *8          LATITUDE
      real *8          LONGITUDE
      integer *4       BOTTOM_DEPTH
      integer *4       MAX_OBSE_DEPTH
      integer *4       NUMBER_OBSE
      integer *4       MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4        ID
      integer*4        Schlitzer_Station_ID
      real*8           TEMPERATURE
      real*8           SALINITY
      real*8           OXYGEN
      real*8           PHOSPHATE
      real*8           SILICATE
      real*8           NITRATE
      integer*4        DEPTH
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                 err_handler
      EXTERNAL                 msg_handler

      INTEGER*4                  login,
      1                          dbproc,
      1                          return_code,
      1                          no_echo,
      1                          lun,
      1                          ipb,
      1                          id_stat,
      1                          id_data,
      1                          leap_year,
      1                          monat,
      1                          i

      character*4                Jahr
      character*2                Tag,
      1                          Stunde,
```

```
      1                        Minute
      character*3              month(12)

      character*30             ASCII_TIME

     INTEGER*4                error
     CHARACTER*(256)          cmdbuf

      CHARACTER*20 password

      REAL*8  ongitud,
      1     atitud

      INTEGER*4  nyear,
      1          nmo,
      1          nda,
      1          nho,
      1          nde,
      1          mode,
      1          nz,
      1          msq,
      1          ni,
      1          nmin,
      1          nseq,
      1          nc,
      1          ns

     character file1*50
C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nmin - minute
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

      DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
      2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C       *****Eroeffnen der Datenbank
C
```

```
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

c
c       no formats used
c

      call lib$get_lun(lun)
      open(unit=lun,
     * file='oth$daten:[socean.schlitzer]schlitzerint.dat',
     * status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Schlitzer_Station_Id#) from Schlitzer_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = statStrt
        end if

        call fdbfcmd(dbproc,
        1  'select max(Schlitzer_Standard_Data_Id#)')
        call fdbfcmd(dbproc,
        1  ' from Schlitzer_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = datStrt
        end if

222   continue
      read(lun,*,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda,
     * nho, nmin, nde, mode, nz, msq

      read(lun,*,end=333) ni

C
C       Die Schlitzer Daten enthalten komplette Jahreszahlen.
C

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 0) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute = '00'
```

```
C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'
         Minute = '00'

        ELSE

         WRITE (TAG,'(I2)') nda
         WRITE (JAHR,'(I4)') nyear
         IF (nho .eq. 24) THEN
           Stunde ='23'
          ELSE
           IF (nho .gt. 24) THEN
             STUNDE = '00'
            ELSE
             WRITE (STUNDE,'(I2)') nho
           END IF
         ENDIF
         IF (nmin .gt. 59) THEN
           Minute = '00'
          ELSE
           WRITE (Minute,'(I2)') nmin
         ENDIF
         MONAT=nmo
        END IF

        ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':'//Minute//'"'

C       ***Speicherung der Stationsdaten****************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into Schlitzer_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)
```

```
C       ********Speicherung der Messdaten*******

        do i=1,ni
        read(lun,*) DATA.DEPTH,
        1             DATA.TEMPERATURE,
        1             DATA.SALINITY,
        1             DATA.OXYGEN,
        1             DATA.PHOSPHATE,
        1             DATA.SILICATE,
        1             DATA.NITRATE

        id_data=id_data+1
        DATA.ID=id_data
        DATA.Schlitzer_STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into Schlitzer_Standard_Data values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.Schlitzer_STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f,', DATA.OXYGEN)
        call fdbfcmd(dbproc,' %f,', DATA.PHOSPHATE)
        call fdbfcmd(dbproc,' %f,', DATA.SILICATE)
        call fdbfcmd(dbproc,' %f)', DATA.NITRATE)
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO

        GOTO 222
333     CONTINUE
        TYPE *,'end of file'
        TYPE *,' there are ',ID_STAT, ' stations in the file'
        CLOSE(LUN)
        call fdbexit()
        END
```

```
      program READSCHLITZER

C     V.Guretsky, AWI, DEC 1991
C
      real*4 t(500), s(500), ox(500),z(500),NO3(500),PO4(500),SI(500)
C
      character file1*50, file2*50,
     *filename*50
C
      integer*4 ncruise
      open(21,file='schlitzerint2.dat',status='old')
C
  222 continue
      read(21,*,end=333) nseq,NCRUISE,nstat, Alon,Alat,
     * nyear,month,nday,
     *nhour,nmin,ndepth,modepth,Nobs,MSQ
C
          type100,nseq,ncruise,nstat,alon,alat,nyear,month,nday,nhour
     *           ,nmin,ndepth,modepth,Nobs,MSQ
      read(21,*)nstlev
          type100,nstlev
C
      do kk=1,NSTLEV
      read(21,*) z(kk), t(kk), s(kk),ox(kk),po4(kk),si(kk),no3(kk)
      type101,z(kk),t(kk),s(kk),ox(kk),po4(kk),si(kk),no3(kk)
      end do
      nsum=nsum+1
      go to 222
  333 continue
  100 format(2x,i3,i6,i6,2f9.2,7i5,2i4)
  101 format(2x,f5.0,6f8.2)
      close(unit=21)
      type*,'number of stations=',nsum
      stop '***END***'
      end
```

for - 17

```
      program BSHR
C V.Guretsky, AWI, Feb 1992
C
C
      real*4 zz(500),
     *       tst(42), sst(42), oxst(42), zst(42),
     *       PST(42),SIST(42),AZOTST(42)
C
      integer*4 crunu,stnum
C*****************************************
C
      open(22,file='intbsh.dat',status='old')
  222 continue
  301 format(2x,f9.4,1x,f9.4)
  302 format(2x,5i7)
  344 format(2x,i3,2x,a4,i4,i4)
501   format (1x,f5.0,6(1x,f10.3))
CWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
      read(22,344,end=333) iseq,a4
      read(22,302) crunu,stnum
      read(22,301) xlon,xlat
      read(22,302) iday,imon,iyear,ihour,imin
      read(22,302) imaxod,nobs,iwmosq,ibot
      read(22,302) mmax
C
      do 11 k=1,mmax
   11 read(22,501) zst(k), tst(k), sst(k), oxst(k),pst(k),sist(k),
     *azotst(k)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      go to 222
  333 continue
      type*,'nstat=',iseq
      close(22)
c
c ISEQ: sequntial number in the current file
c A4: NODC code
c XLON: longitude in decimal degrees
c XLAT: latitude in decimal degrees
c IDAY: day of observation
c IMON: month of observation
c IYEAR: year of observation
c IHOUR: hour of observation
c IMIN: minute of observation
c IBOT: bottom depth
c IMAXOD: maximum observed depth
c NOBS: number of observed depths
c IWMOSQ: ten degree WMO square number
c DEPTH, TEMPERATURE, SALINITY, OXYGENE, PHOSPHATE, SILICATE, NITRATE
C
      stop '***END***'
       end
```

```
      options /check=all
      program AWIload

C     CREATOR::M. Reinke
C     CREA_DATE::25-Jul-1990
C     CHANGES:: 1991-10-08 L.-P. Kurdelski
C                          reading AWI data
C
      structure /station/
      integer *4       ID
      integer *4       CRUISE_NUMBER
      integer *4       STATION_NUMBER
      real *8          LATITUDE
      real *8          LONGITUDE
      integer *4       BOTTOM_DEPTH
      integer *4       MAX_OBSE_DEPTH
      integer *4       NUMBER_OBSE
      integer *4       MARSDEN_SQUARE
      end structure

      structure /data/
      integer*4        ID
      integer*4        AWI_Station_ID
      real*8           TEMPERATURE
      real*8           SALINITY
      integer*4        DEPTH
      end structure

      record /STATION/ STATION
      record /DATA/ DATA

      include '(fsybdb)'
      include '($smgdef)'
      include '($ttdef)'
      include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                 err_handler
      EXTERNAL                 msg_handler

        PARAMETER (paramstat = 20000000)
        PARAMETER (paramdat = 200000000)

        INTEGER*4                login,
        1                        dbproc,
        1                        return_code,
        1                        no_echo,
        1                        lun,
        1                        lun1,
        1                        ipb,
        1                        id_stat,
        1                        id_data,
        1                        leap_year,
        1                        monat,
        1                        i,
        1                        index,
        1                        KK

        character*4              Jahr
        character*2              Tag,
        1                        Stunde,
        1                        Minute
        character*3              month(12)
```

```
      character*80            name

      character*30            ASCII_TIME

      INTEGER*4               error
      CHARACTER*(256)         cmdbuf

      CHARACTER*20 password

      REAL*8  ongitud,
     1     atitud

      INTEGER*4  nyear,
     1           nmo,
     1           nda,
     1           nho,
     1           nde,
     1           mode,
     1           nz,
     1           msq,
     1           ni,
     1           nmin,
     1           nseq,
     1           nc,
     1           ns

      character file1*50
C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nmin - minute
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

      DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
     2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
```

```
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

c       Reading nseq, nc, ns, msq
  500 format(2x,i12)
c       Reading lat, lon
  501 format(2x,f11.5,f15.5)
c       Reading day, month, year, hour, min
  502 format(2x,5i12)
c       Reading mode, nde, ni, obs
  503 format(2x,2i12)
c       Reading depth, temperature, salinity, oxygen
  102 format(2x,i4,1x,3f8.3)

c
c       Dateiliste oeffnen
c
      call lib$get_lun(lun1)
      open(unit=lun1,
     * file='sys$user:[kurdelski.southernocean.for]awiliste.dat',
     * status='old')

       do index = 1, 1000
         read(lun1,'(a)',err=3,end=3) name

      call lib$get_lun(lun)
      open(unit=lun,
     * file=name,
     * status='old')
        TYPE *,'begin of file'

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(AWI_Station_Id#) from AWI_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = paramstat
        end if

        call fdbfcmd(dbproc,
        1  'select max(AWI_Standard_Data_Id#)')
        call fdbfcmd(dbproc,
        1  ' from AWI_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = paramdat
        end if

  222 continue

C      read(lun,500,end=333) nseq
C      read(lun,500,end=333) nc
C      read(lun,500,end=333) ns
C      read(lun,501,end=333) atitud, ongitud
C      read(lun,502,end=333) nda, nmo, nyear, nho, nmin
C      read(lun,503,end=333) nde, mode
```

```
C      read(lun,503,end=333) nz, ni
C      read(lun,500,end=333) msq

       read(lun,*,end=333) nseq
       read(lun,*,end=333) nc
       read(lun,*,end=333) ns
       read(lun,*,end=333) atitud, ongitud
       read(lun,*,end=333) nda, nmo, nyear, nho, nmin
       read(lun,*,end=333) nde, mode
       read(lun,*,end=333) nz, ni
       read(lun,*,end=333) msq


C
C       Die AWI Daten enthalten komplette Jahreszahlen.
C

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute = '00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'
         Minute = '00'

        ELSE

         WRITE (TAG,'(I2)') nda
         WRITE (JAHR,'(I4)') nyear
         IF (nho .eq. 24) THEN
           Stunde ='23'
          ELSE
           IF (nho .gt. 24) THEN
             STUNDE = '00'
            ELSE
             WRITE (STUNDE,'(I2)') nho
           END IF
         ENDIF
         IF (nmin .gt. 59) THEN
           Minute = '00'
          ELSE
           WRITE (Minute,'(I2)') nmin
         ENDIF
         MONAT=nmo
        END IF
```

```
      ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
      2//STUNDE//':'//Minute//'"'

C     ***Speicherung der Stationsdaten****************

      ID_STAT=ID_STAT+1
      STATION.ID=ID_STAT
      STATION.CRUISE_NUMBER=nc
      STATION.STATION_NUMBER=ns
      STATION.LATITUDE=atitud
      STATION.LONGITUDE=ongitud
      STATION.BOTTOM_DEPTH=nde
      STATION.MAX_OBSE_DEPTH=mode
      STATION.NUMBER_OBSE=nz
      STATION.MARSDEN_SQUARE=msq

      type *, station.id,' ',ascii_time

      call fdbcmd(dbproc,' insert into AWI_Station values ( ')
      call fdbfcmd(dbproc,' %d,', STATION.ID)
      call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
      call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
      call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
      call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
      call fdbfcmd(dbproc,' %s,', ASCII_TIME)
      call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
      call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)
C     ********Speicherung der Messdaten*******

      do i=1,ni
C     read(lun,102) DATA.DEPTH,
C     1             DATA.TEMPERATURE,
C     1             DATA.SALINITY,
C     1             DATA.OXYGEN
      read(lun,*) KK,
      1             DATA.DEPTH,
      1             DATA.TEMPERATURE,
      1             DATA.SALINITY

      id_data=id_data+1
      DATA.ID=id_data
      DATA.AWI_STATION_ID = STATION.ID

      call fdbcmd(dbproc,' insert into AWI_Standard_Data values (')
      call fdbfcmd(dbproc,' %d,', DATA.ID)
      call fdbfcmd(dbproc,' %d,', DATA.AWI_STATION_ID)
      call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
      call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
      call fdbfcmd(dbproc,' %f)', DATA.SALINITY)
      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)

      END DO

      GOTO 222
333   CONTINUE
      TYPE *,'end of file'
      TYPE *,' there are ',ID_STAT, ' stations in the file'
      CLOSE(LUN)
      end do
```

```
3       continue
        close(LUN1)

        call fdbexit()
        END
```

for-19

```
options /check=all

C       CREATOR::M. REINKE
C       CREA_DATE::25-Jul-1990
C       CHANGES:: 1994-05-09 BM
C                                reading AARI-LDGO-DATA
C
        structure /station/
        integer *4      ID
        integer *4      CRUISE_NUMBER
        integer *4      STATION_NUMBER
        real *8         LATITUDE
        real *8         LONGITUDE
        integer *4      BOTTOM_DEPTH
        integer *4      MAX_OBSE_DEPTH
        integer *4      NUMBER_OBSE
        integer *4      MARSDEN_SQUARE
        end structure

        structure /data/
        integer*4       ID
        integer*4       Station_ID
        real*8          TEMPERATURE
        real*8          SALINITY
        real*8          OXYGEN
        integer*4       DEPTH
        real*8          PHOSPHATE
        real*8          SILICATE
        real*8          NITRAT
        end structure

        record /STATION/ STATION
        record /DATA/ DATA

        include '(fsybdb)'
        include '($smgdef)'
        include '($ttdef)'
        include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                err_handler
      EXTERNAL                msg_handler

        INTEGER*4               login,
        1                       dbproc,
        1                       return_code,
        1                       no_echo,
        1                       lun,
        1                       ipb,
        1                       id_stat,
        1                       id_data,
        1                       leap_year,
        1                       monat,
        1                       i

        character*4             Jahr
        character*2             Tag,
        1                       Stunde,
        1                       Minute
        character*3             month(12)

        character*30            ASCII_TIME

      INTEGER*4               error
      CHARACTER*(256)         cmdbuf


      CHARACTER*20 password

C-------Victors Data and Formats
```

```
C       read interpolated AARI-ldgo data
C
C       V.Guretsky, AWI, Dec 1993
C
C
       character file1*80, file2*80
C
      real*4    zg1(900),tg1(900),sg1(900),og1(900),zst(42),
     *       fob1(900), zob1(900) ,TST(42),SST(42),OST(42)
C
      integer*4 CRUNU

      integer*4 nyear, nmonth, nday, nhour
C
      data zst /0.,10.,20.,30.,50.,75.,100.,125.,150.,200.,
     * 250.,300.,350.,400.,500.,600.,700.,750.,800.,900.,
     * 1000.,1100.,1200.,1300.,1400.,1500.,1750.,2000.,2250.,2500.,
     * 2750.,3000.,3250.,3500.,3750.,4000.,4500.,5000.,5500.,6000.,
     * 6500.,7000./
C
C     ----------------------------------


        DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
        2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C


C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

       call lib$get_lun(lun)
       open(unit=lun, file='oth$daten:[socean.aari3]aarildgo.dat',
     * status ='old')


        call fdbfcmd(dbproc,
        1       'select max(AariLdgo_Station_Id#) from AariLdgo_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = 3200000
        end if

        call fdbfcmd(dbproc,
        1  'select max(AariLdgo_Standard_Data_Id#)
        1  from AariLdgo_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)
```

```
if (ID_DATA .eq. 0) then
           ID_DATA = 320000000
        end if

C       ******** reading data from disk *********
C
C
C     nseq - sequential number of station in the file
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nde - Bottom_Depth
C     mode - Max_Obse_pressure bzw. _depth
C     nz - number of observed levels for the cast
C     mst - number of interpolated (Standard) levels covered by the cast
C     msq - Marsden Square
C

C
401   format(2x,3I7,2x,2f9.4,2x,8i5)
402   format(2x,f7.2,1x,3f8.3)

222   continue
      read(lun,401,end=333) nseq, CRUNU, ns, ongitud, atitud, nyear,
     * nmonth, nday,
     * nhour,
     * nde, mode, nz, msq
      read(lun,401) mst
C
C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********
C-------BM Jahresangaben in *.dat sind von der Form "84" statt "1984"
        nyear = nyear + 1900
        leap_year = mod(nyear, 4)

      if (((nhour.gt.24 .and. nhour.ne.99) .or. nhour.lt.0) .OR.
     *   (nday.gt.31 .or. nday.lt.1 ) .OR.
     *   (nmonth.gt.12 .or. nmonth.lt.1)  .OR.
     *   (nyear.gt.1994 .or. nyear.lt.1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute = '00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nday .eq.29 .and.
        1        nmonth .eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'
         Minute ='00'
        ELSE

         WRITE (TAG,'(I2)') nday
         WRITE (JAHR,'(I4)') nyear
         IF (nhour .eq. 24) THEN
          Stunde ='23'
         ELSE
          IF (nhour .eq. 99) THEN
              STUNDE = '00'
            ELSE
              WRITE (STUNDE,'(I2)') nhour
```

```
          END IF
         END IF
         IF (nmin .eq. 99) THEN
             Minute = '00'
           ELSE
             WRITE (Minute,'(I2)') nmin
         END IF
         MONAT=nmonth
        END IF

        ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':'//Minute//'"'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=CRUNU
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into AariLdgo_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

C       ********Speicherung der Messdaten*******
      do  k=1,mst
      read(lun,102) zst(k), tst(k), sst(k), OST(K)
C


           DATA.DEPTH = zst(k)
           DATA.TEMPERATURE = tst(k)
           DATA.SALINITY = sst(k)
           DATA.OXYGEN = ost(k)

        id_data=id_data+1
        DATA.ID=id_data
        DATA.STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into AariLdgo_Standard_Data')
        call fdbfcmd(dbproc,' values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f)', DATA.OXYGEN )
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO
        GOTO 222
333   continue
```

```
      type *,'end of file'
      type*,'total of ',nseq
      close(lun)
      call fdbexit()
      END
```

for-20

```
options /check=all
        program ozedb_load

C       CREATOR::M. Reinke
C       CREA_DATE::25-Jul-1990
        structure /station/
        integer *4      ID
        integer *4      CRUISE_NUMBER
        integer *4      STATION_NUMBER
        real *8         LATITUDE
        real *8         LONGITUDE
        integer *4      BOTTOM_DEPTH
        integer *4      MAX_OBSE_DEPTH
        integer *4      NUMBER_OBSE
        integer *4      MARSDEN_SQUARE
        end structure

        structure /data/
        integer*4       ID
        integer*4       AARI_Station_ID
        real*8          TEMPERATURE
        real*8          SALINITY
        real*8          OXYGEN
        integer*4       DEPTH
        end structure

        record /STATION/ STATION
        record /DATA/ DATA

        include '(fsybdb)'
        include '($smgdef)'
        include '($ttdef)'
        include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                err_handler
      EXTERNAL                msg_handler

        INTEGER*4               login,
        1                       dbproc,
        1                       return_code,
        1                       no_echo,
        1                       lun,
        1                       ipb,
        1                       id_stat,
        1                       id_data,
        1                       leap_year,
        1                       monat,
        1                       i

        character*4             Jahr
        character*2             Tag,
        1                       Stunde
        character*3             month(12)

        character*30            ASCII_TIME

      INTEGER*4                 error
      CHARACTER*(256)           cmdbuf


        CHARACTER*20 password

        INTEGER*4  nseq,
        1     nc,
        1     ns

        REAL*8  ongitud,
        1     atitud
```

```
INTEGER*4  nyear,
     1          nmo,
     1          nda,
     1          nho,
     1          nde,
     1          mode,
     1          nz,
     1          msq,
     1          ni

      character file1*50

C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
      nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

        DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
        2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

C     Guretsky, AWI, 21 June 1990
C
  101 format(2x,3i7,2f8.2,9i7)
  102 format(2x,i4,x,3f8.3)

15    format(' Name of the input file: '$)
20    format(a50)
      type 15
      accept 20, file1
      call lib$get_lun(lun)
      open(unit=lun, file=file1,status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
```

```
1       'select max(Aari_Station_Id#) from Aari_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        call fdbfcmd(dbproc,
        1  'select max(Aari_Standard_Data_Id#) from Aari_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

  222 continue
      read(lun,101,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda, nho, nde, mode, nz, msq

      read(lun,101) ni

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1989 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'

        ELSE

         WRITE (TAG,'(I2)') nda
         WRITE (JAHR,'(I4)') nyear
         IF (nho .eq. 24) THEN
          Stunde ='23'
         ELSE
          WRITE (STUNDE,'(I2)') nho
         END IF
         MONAT=nmo
        END IF

        ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':00'//'"'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq
```

```
      type *, station.id,' ',ascii_time

      call fdbcmd(dbproc,' insert into Aari_Station values ( ')
      call fdbfcmd(dbproc,' %d,', STATION.ID)
      call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
      call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
      call fdbfcmd(dbproc,' %s,', ASCII_TIME)
      call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
      call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
      call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
      call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
      call fdbfcmd(dbproc,' %d,', STATION.MARSDEN_SQUARE)
      call fdbcmd(dbproc,'0,0)')

      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)
C     ********Speicherung der Messdaten*******

      do i=1,ni
      read(lun,102) DATA.DEPTH,
     1              DATA.TEMPERATURE,
     1              DATA.SALINITY,
     1              DATA.OXYGEN

      id_data=id_data+1
      DATA.ID=id_data
      DATA.AARI_STATION_ID = STATION.ID

      call fdbcmd(dbproc,' insert into Aari_Standard_Data values (')
      call fdbfcmd(dbproc,' %d,', DATA.ID)
      call fdbfcmd(dbproc,' %d,', DATA.AARI_STATION_ID)
      call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
      call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
      call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
      call fdbfcmd(dbproc,' %f,', DATA.OXYGEN )
      call fdbcmd(dbproc,' 0,0)')
      call fdbsqlexec(dbproc)
      return_code = fdbresults(dbproc)

      END DO

      GOTO 222
333   CONTINUE
      TYPE *,'end of file'
      TYPE *,' there are ',ID_STAT, ' stations in the file'
      CLOSE(LUN)
      call fdbexit()
      END
```

for-21

```
options /check=all
        program Argentineload

C       CREATOR::M. Reinke
C       CREA_DATE::25-Jul-1990
C       CHANGES:: 1991-10-08 L.-P. Kurdelski
C                            reading Argentine data
C                 1991-11-06 L.-P. Kurdelski
C                            changeing the datafile name
C
        structure /station/
        integer *4      ID
        integer *4      CRUISE_NUMBER
        integer *4      STATION_NUMBER
        real *8         LATITUDE
        real *8         LONGITUDE
        integer *4      BOTTOM_DEPTH
        integer *4      MAX_OBSE_DEPTH
        integer *4      NUMBER_OBSE
        integer *4      MARSDEN_SQUARE
        end structure

        structure /data/
        integer*4       ID
        integer*4       Argentine_Station_ID
        real*8          TEMPERATURE
        real*8          SALINITY
        real*8          OXYGEN
        integer*4       DEPTH
        end structure

        record /STATION/ STATION
        record /DATA/ DATA

        include '(fsybdb)'
        include '($smgdef)'
        include '($ttdef)'
        include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                err_handler
      EXTERNAL                msg_handler

        INTEGER*4               login,
        1                       dbproc,
        1                       return_code,
        1                       no_echo,
        1                       lun,
        1                       ipb,
        1                       id_stat,
        1                       id_data,
        1                       leap_year,
        1                       monat,
        1                       i

        character*4             Jahr
        character*2             Tag,
        1                       Stunde,
        1                       Minute
        character*3             month(12)

        character*30            ASCII_TIME

      INTEGER*4               error
      CHARACTER*(256)         cmdbuf

        CHARACTER*20 password

        REAL*8  ongitud,
        1     atitud
```

```
        INTEGER*4  nyear,
        1          nmo,
        1          nda,
        1          nho,
        1          nde,
        1          mode,
        1          nz,
        1          msq,
        1          ni,
        1          nmin,
        1          nseq,
        1          nc,
        1          ns

      character file1*50
C
C     nseq - sequential number of station in the file
C     nc - cruise number
C     ns - station_number
C     ongitud - Longitude
C     atitude - Latitude
C     nyear - Year
C     nmo - month
C     nda - day
C     nho - hour
C     nmin - minute
C     nde - Bottom_Depth
C     mode - Max_Obse_Depth
C     nz - number_obse
C     msq - Marsden_Square
C     ni - number of standard (interpolated) levels
C

        DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
        2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
      to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

c       ******** reading data from disk *********

c       Reading nseq, nc, ns, lon, lat
  400 format(2x,3i7,2f8.2)
c       Reading ny, nm, nd, nh, nm, nde, mode, nobs, msq
  401 format(2x,9i7)
c       Reading nmax
  402 format(2x,i3)
c       Reading depth, temperature, salinity, oxygen
  102 format(2x,i4,1x,3f8.3)

      call lib$get_lun(lun)
      open(unit=lun,
     * file='oth$daten:[socean.argent]interarg4.dat',
     * status='old')
```

```
C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Argentine_Station_Id#) from Argentine_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = 3000000
        end if

        call fdbfcmd(dbproc,
        1  'select max(Argentine_Standard_Data_Id#)')
        call fdbfcmd(dbproc,
        1  ' from Argentine_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = 30000000
        end if

  222 continue
      read(lun,400,end=333) nseq, nc, ns, ongitud, atitud
      read(lun,401,end=333) nyear, nmo, nda, nho, nmin,
     * nde, mode, nz, msq

      read(lun,402,end=333) ni
C
C       Die Argentine Daten enthalten komplette Jahreszahlen.
C

C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if ((nho.gt.24 .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute = '00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'
         Minute = '00'

        ELSE

         WRITE (TAG,'(I2)') nda
         WRITE (JAHR,'(I4)') nyear
         IF (nho .eq. 24) THEN
           Stunde ='23'
          ELSE
```

```
      IF (nho .gt. 24) THEN
            STUNDE = '00'
           ELSE
            WRITE (STUNDE,'(I2)') nho
          END IF
         ENDIF
         IF (nmin .gt. 59) THEN
           Minute = '00'
          ELSE
           WRITE (Minute,'(I2)') nmin
         ENDIF
         MONAT=nmo
        END IF

        ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':'//Minute//'"'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into Argentine_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)
C       ********Speicherung der Messdaten*******

        do i=1,ni
        read(lun,102) DATA.DEPTH,
        1             DATA.TEMPERATURE,
        1             DATA.SALINITY,
        1             DATA.OXYGEN

        id_data=id_data+1
        DATA.ID=id_data
        DATA.Argentine_STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into Argentine_Standard_Data values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.Argentine_STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f)', DATA.OXYGEN)
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO

        GOTO 222
333     CONTINUE
```

```
TYPE *,'end of file'
       TYPE *,' there are ',ID_STAT, ' stations in the file'
       CLOSE(LUN)
       call fdbexit()
       END
```

for-22

```
      program READJARE
C V.Guretsky, AWI, 15 APRIL 1991
C
      integer*4 crunu, numstat
      real*4 tem(42), sal(42), oxy(42), po(42), si(42), n3(42), zz(42)
C
      open(unit=21,file='oth$daten:[socean.jare]jareall.dat'
     *,status='old')
C         I N P U T
      do 333 L=1,119
      read(21,202) nseq,CRUNU,numstat,A,P,nyear,month,nday,
     *nhour,minut,ndep,modepth,n,msq
C
      type 202, nseq,CRUNU,numstat,A,P,nyear,month,nday,
     *nhour,minut,ndep,modepth,n,msq
C
      read(21,102) mmax
      type 102, mmax
  102 format(2x,i3)
C
      do 2 k=1,mmax
      read(21,103) zz(k),tem(k),sal(k),oxy(k),PO(k),n3(k),SI(k)
    2 type 103,  zz(k), tem(k), sal(k), oxy(k),PO(k),N3(k),SI(k)
C

C      VARIABLES:
C NSEQ - sequential number of station in the file
C CRUNU - Cruise Number
C NUMSTAT - Station Number
C A - Longitude
C P - Latitude
C NYEAR - Year
C MONTH - month
C NDAY - Day
C NHOUR - Hour
C MINUT Minutes
C NDEP - BNottom Depth
C MODEPH - Max_Obse_Depth
C N - Number_Obse
C MMAX-Number of interpolated levels
C ZZ - Depth in meters
C TEM - temperature
C SAL - salinity
C OXY - Oxygen
C PO - Phosphatus
C N3 - Nitrate
C SI - Silicate
C
  103 format(2x,f5.0,6f8.3)
  202 format(2x,3i7,2f8.2,9i7)
C  ------------------------------------------------------------
  333 continue
      close (unit=21)
      stop '***END***'
       end
```

```
      program readmuin
Cread interpolated MUENCH DATA
C
C     V.Guretsky, AWI, JUNE 1991
C
      character file1*64
C
      integer*4 NCRU
C
      real*4    zg1(5000),tg1(5000),sg1(5000),zst(42),
     *       fob1(5000), zob1(5000) ,TST(42),SST(42)
C
      data zst /0.,10.,20.,30.,50.,75.,100.,125.,150.,200.,
     * 250.,300.,350.,400.,500.,600.,700.,750.,800.,900.,
     * 1000.,1100.,1200.,1300.,1400.,1500.,1750.,2000.,2250.,2500.,
     * 2750.,3000.,3250.,3500.,3750.,4000.,4500.,5000.,5500.,6000.,
     * 6500.,7000./
C     ------------------------------------
  100 format(a64)
C
       type*, 'Name of input file'
       accept 100,file1
       open(unit=20, file=file1,status='old')
C
C-------------------------------------------------
  222 continue
      read(20,202,end=333) nseq,NCRU,numst, ongitud,atitud
      read(20,203) nyear,nmonth,nday,
     *nhour,nmin,ndepth,modepth,nlev,msq
  104 format(5(1x,f7.2,2f7.3))
  202 format(2x,3i7,2f8.2)
  203 format(10i7)
   22 format(2x,i3,2x,f6.1,2f7.3)
      read(20,22) J
C
      do9 i=1,J
      read(20,22) ii, zg1(i),Tg1(i),Sg1(i)
    9 continue
C============================================================
      go to 222
  333 continue
C
      type*,'total number of stations in the file is ',nseq
      close(unit=20)
      stop '********* E N D *********'
      END
```

```
options /check=all

C       CREATOR::M. REIKNE
C       CREA_DATE::25-Jul-1990
C       CHANGES:: 1991-02-13 L.-P. Kurdelski
C                            reading Tkyo Fisheries data
C
        structure /station/
        integer *4      ID
        integer *4      CRUISE_NUMBER
        integer *4      STATION_NUMBER
        real *8         LATITUDE
        real *8         LONGITUDE
        integer *4      BOTTOM_DEPTH
        integer *4      MAX_OBSE_DEPTH
        integer *4      NUMBER_OBSE
        integer *4      MARSDEN_SQUARE
        end structure

        structure /data/
        integer*4       ID
        integer*4       Station_ID
        real*8          TEMPERATURE
        real*8          SALINITY
        real*8          OXYGEN
        integer*4       DEPTH
        end structure

        record /STATION/ STATION
        record /DATA/ DATA

        include '(fsybdb)'
        include '($smgdef)'
        include '($ttdef)'
        include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                err_handler
      EXTERNAL                msg_handler

        INTEGER*4               login,
        1                       dbproc,
        1                       return_code,
        1                       no_echo,
        1                       lun,
        1                       ipb,
        1                       id_stat,
        1                       id_data,
        1                       leap_year,
        1                       monat,
        1                       i

        character*4             Jahr
        character*2             Tag,
        1                       Stunde,
        1                       Minute
        character*3             month(12)

        character*30            ASCII_TIME

      INTEGER*4               error
      CHARACTER*(256)         cmdbuf


        CHARACTER*20 password

        INTEGER*4  nseq,
        1     nc,
        1     ns
```

```
      REAL*8  ongitud,
     1        atitud

              INTEGER*4  nyear,
     1                   nmo,
     1                   nda,
     1                   nho,
     1                   nmin,
     1                   nde,
     1                   mode,
     1                   nz,
     1                   msq,
     1                   ni

            character file1*50

C
C           nseq - sequential number of station in the file
C           nc - cruise number
C           ns - station_number
C           ongitud - Longitude
C           atitude - Latitude
C           nyear - Year
C           nmo - month
C           nda - day
C           nho - hour
C           nmin - minute
C           nde - Bottom_Depth
C           mode - Max_Obse_Depth
C           nz - number_obse
C           msq - Marsden_Square
C           ni - number of standard (interpolated) levels
C

              DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
     2                    'Sep','Oct','Nov','Dec'/

C
C           Install the user-supplied error-handling and message-handling
C           routines. They are defined at the bottom of this source file.
C
            call fdberrhandle(err_handler)
            call fdbmsghandle(msg_handler)
C
C           Allocate and initialize the LOGINREC record to be used
            to open a connection to the DataServer.
C
            login = fdblogin()
            call fdbsetluser(login, 'sa')
            call ask_for_pw(password)
            call fdbsetlpwd(login, password)
C
C
C

C             *****Eroeffnen der Datenbank
C
            dbproc = fdbopen(login, NULL)
            call fdbuse(dbproc,'SouthernOceanDB')

c             ******** reading data from disk *********

C           Guretsky, AWI, 21 June 1990
C
  401       format(2x,3i7,2f8.2,9i7)
  102       format(2x,i4,1x,3f8.3)

15          format(' Name of the input file: '$)
20          format(a50)
            type 15
            accept 20, file1
            call lib$get_lun(lun)
```

```
      open(unit=lun, file=file1,status='old')

C       *****Zaehlung der  Records

        call fdbfcmd(dbproc,
        1       'select max(Station_Id#) from Tokyo_Fisheries_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = 500000
        end if

        call fdbfcmd(dbproc,
        1  'select max(Standard_Data_Id#) from Tokyo_Fisheries_Standard_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = 5000000
        end if

  222 continue
      read(lun,401,end=333) nseq, nc, ns, ongitud, atitud,
     * nyear, nmo, nda, nho, nmin, nde, mode, nz, msq

      read(lun,401) ni

C
C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyear,4)

        if (((nho.gt.24 .and. nho.ne.99) .or. nho .lt. 00) .OR.
        1   (nda.gt.31 .or. nda .lt. 1 ) .OR.
        1   (nmo.gt.12 .or. nmo .lt. 1)  .OR.
        1   (nyear.gt.1990 .or. nyear .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute = '00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nda.eq.29 .and.
        1        nmo.eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'
         Minute ='00'
        ELSE

         WRITE (TAG,'(I2)') nda
         WRITE (JAHR,'(I4)') nyear
         IF (nho .eq. 24) THEN
          Stunde ='23'
         ELSE
          IF (nho .eq. 99) THEN
              STUNDE = '00'
            ELSE
              WRITE (STUNDE,'(I2)') nho
          END IF
```

```
 END IF
         IF (nmin .eq. 99) THEN
             Minute = '00'
           ELSE
             WRITE (Minute,'(I2)') nmin
         END IF
         MONAT=nmo
        END IF

        ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':'//Minute//'"'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=nc
        STATION.STATION_NUMBER=ns
        STATION.LATITUDE=atitud
        STATION.LONGITUDE=ongitud
        STATION.BOTTOM_DEPTH=nde
        STATION.MAX_OBSE_DEPTH=mode
        STATION.NUMBER_OBSE=nz
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into Tokyo_Fisheries_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)
C       ********Speicherung der Messdaten*******

        do i=1,ni
        read(lun,102) DATA.DEPTH,
        1             DATA.TEMPERATURE,
        1             DATA.SALINITY,
        1             DATA.OXYGEN

        id_data=id_data+1
        DATA.ID=id_data
        DATA.STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into Tokyo_Fisheries_Standard_Data')
        call fdbfcmd(dbproc,' values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f)', DATA.OXYGEN )
        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO

        GOTO 222
333     CONTINUE
        TYPE *,'end of file'
        TYPE *,' there are ',ID_STAT, ' stations in the file'
        CLOSE(LUN)
        call fdbexit()
```

END

for-23

```
      program readmuin
Cread interpolated MUENCH DATA
C
C     V.Guretsky, AWI, JUNE 1991
C
      character file1*64
C
      integer*4 NCRU
C
      real*4    zg1(5000),tg1(5000),sg1(5000),zst(42),
     *       fob1(5000), zob1(5000) ,TST(42),SST(42)
C
      data zst /0.,10.,20.,30.,50.,75.,100.,125.,150.,200.,
     * 250.,300.,350.,400.,500.,600.,700.,750.,800.,900.,
     * 1000.,1100.,1200.,1300.,1400.,1500.,1750.,2000.,2250.,2500.,
     * 2750.,3000.,3250.,3500.,3750.,4000.,4500.,5000.,5500.,6000.,
     * 6500.,7000./
C     ------------------------------------
  100 format(a64)
C
       type*, 'Name of input file'
       accept 100,file1
       open(unit=20, file=file1,status='old')
C
C-----------------------------------------------
  222 continue
      read(20,202,end=333) nseq,NCRU,numst, ongitud,atitud
      read(20,203) nyear,nmonth,nday,
     *nhour,nmin,ndepth,modepth,nlev,msq
  104 format(5(1x,f7.2,2f7.3))
  202 format(2x,3i7,2f8.2)
  203 format(10i7)
   22 format(2x,i3,2x,f6.1,2f7.3)
      read(20,22) J
C
      do9 i=1,J
      read(20,22) ii, zg1(i),Tg1(i),Sg1(i)
    9 continue
C=============================================================
      go to 222
  333 continue
C
      type*,'total number of stations in the file is ',nseq
      close(unit=20)
      stop '********* E N D *********'
      END
```

for-24

```
      program argnewcrnum
C   this program makes new Cruise numbers
C     V.Guretsky, AWI, August 1991
C
      real*4 tem(2000), sal(2000), oxy(2000),z(2000)
      character file1*15, file2*15, country*2,ship*2,cruise*3,
     * station*5,Aa*1,AO*1,symbol*1,blank*3,tsym*1,ssym*1
      character*1 shipcruise(5)
      character*2 sa(100)
      integer*4 ncruise
C
  100 format(a15)
      open(unit=22,file='argent2.dat',status='old')
      open(unit=24,file='argent3.dat',status='new')
C
      ns=1
C
  222 continue
C_____________________________inPUT_____________________________
      read(22,202,end=333) nseq,NCRUISE,nstat, ongitud,atitud
      read(22,203) nyear,nmonth,nday,
     *nhour,nmin,ndepth,modepth,K,msq10
      read(22,204)country
      read(22,204)ship
      read(22,205)cruise
  202 format(2x,3i7,2f8.2)
  203 format(10i7)
  204 format(2x,a2)
  205 format(2x,a3)
  245 format(2x,3a1)
      do kk=1,K
      read(22,103) z(kk), tem(kk), sal(kk),oxy(kk)
  103 format(2x,f5.0,2f7.3,f6.2)
      end do
C   ____________________________________________________
      NCRUISE=NCRUISE+58000
C-----------------------------------------------------
      write(24,202) nseq,NCRUISE,nstat, ongitud,atitud
      write(24,203) nyear,nmonth,nday,
     *nhour,nmin,ndepth,modepth,K,msq10
      write(24,204)country
      write(24,204)ship
      write(24,205)cruise
      do kk=1,K
      write(24,103) z(kk), tem(kk), sal(kk),oxy(kk)
      end do
C_____________________________________________________
      go to 222
  333 continue
      close(unit=22)
      close(unit=24)
      type*,'number of stations=',nseq
      stop '***END***'
      end
```

for -25

```
      program readargent
C   this program reads Argentine data
C     V.Guretsky, AWI, June 1991
C
      real*4 tem(42), sal(42), oxy(42),z(42)
      character file1*15, file2*15, country*2,ship*2,cruise*3,
C
      open(22,file='interarg4.dat',status='old')
    2 continue
      read(22,202,end=3) nseq,NCRUISE,nstat, ongitud,atitud
      read(22,203) nyear,month,nday,
     *nhour,nmin,depth,modepth,K,msq10
      read(22,204)country
      read(22,204)ship
      read(22,205)cruise
  202 format(2x,3i7,2f8.2)
  203 format(10i7)
  204 format(2x,a2)
  205 format(2x,a3)
C
      do kk=1,K
      read(22,103) z(kk), tem(kk), sal(kk),oxy(kk)
  103 format(2x,f5.0,2f7.3,f6.2)
      end do
      go to 2
    3 continue
      close(unit=22)
      stop '***END***'
      end
```

fo-26

```
options /check=all

C       CREATOR::M. REINKE
C       CREA_DATE::25-Jul-1990
C       CHANGES:: 1994-05-09 BM
C                                reading BSH2 data
C
        structure /station/
        integer *4      ID
        integer *4      CRUISE_NUMBER
        integer *4      STATION_NUMBER
        real *8         LATITUDE
        real *8         LONGITUDE
        integer *4      BOTTOM_DEPTH
        integer *4      MAX_OBSE_DEPTH
        integer *4      NUMBER_OBSE
        integer *4      MARSDEN_SQUARE
        end structure

        structure /data/
        integer*4       ID
        integer*4       Station_ID
        real*8          TEMPERATURE
        real*8          SALINITY
        real*8          OXYGEN
        integer*4       DEPTH
        real*8          PHOSPHATE
        real*8          SILICATE
        real*8          NITRAT
        end structure

        record /STATION/ STATION
        record /DATA/ DATA

        include '(fsybdb)'
        include '($smgdef)'
        include '($ttdef)'
        include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                err_handler
      EXTERNAL                msg_handler

        INTEGER*4               login,
        1                       dbproc,
        1                       return_code,
        1                       no_echo,
        1                       lun,
        1                       ipb,
        1                       id_stat,
        1                       id_data,
        1                       leap_year,
        1                       monat,
        1                       i

        character*4             Jahr
        character*2             Tag,
        1                       Stunde,
        1                       Minute
        character*3             month(12)

        character*30            ASCII_TIME

      INTEGER*4                 error
      CHARACTER*(256)           cmdbuf


      CHARACTER*20 password

C-------Victors Data and Formats
```

```
C
C DESCRIPTION OF VARIABLES:
C mseq-sequential number of station in BSH2.DAT file
C nstat - Station_Number
C ncruise - Cruise_Number
C nyear - Year
C nmonth - Month
C nday - Day
C nhour - Hour
C nmin - Minutes
C gradlat - Latitude (grad.)
C gradlon - Longitude (grad.)
C ndepth - Bottom_Depth
C modepth - Max_Obse_Depth
C nobs - Number_Obse
C msq - Marsden_Square
C MMAX - Number of Standard Levels
C ZST - array of standard depths, meters
C tst - Temperature array, grad. C
C sst - salinity array
C ost - oxygen array, ml/l
C sist - silicate array, mg_at/l
C fnst - nitrate array, mg_at/l
C phst - phosphate array, mg_at/l
C
      character file1*80, file2*80
C
      real*4    z(900),t(900),s(900),ox(900),si(900),ph(900),fn(900),
     *   zst(42),
     *   fob1(900), zob1(900) ,TST(42),SST(42),OST(42),
     *   phst(42),fnst(42),sist(42)

      integer*4 nyear,nmonth,nday,nhour,nmin
C
      data zst /0.,10.,20.,30.,50.,75.,100.,125.,150.,200.,
     * 250.,300.,350.,400.,500.,600.,700.,750.,800.,900.,
     * 1000.,1100.,1200.,1300.,1400.,1500.,1750.,2000.,2250.,2500.,
     * 2750.,3000.,3250.,3500.,3750.,4000.,4500.,5000.,5500.,6000.,
     * 6500.,7000./
C
  100 format(a80)
C


       DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
       2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C


C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

       call lib$get_lun(lun)
       open(unit=lun, file='oth$daten:[socean.save]BSH2.DAT',
     * status ='old')
```

```
        call fdbfcmd(dbproc,
        1       'select max(BSH2_Station_Id#) from BSH2_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = 3100000
        end if

        call fdbfcmd(dbproc,
        1  'select max(BSH2_Data_Id#) from BSH2_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = 310000000
        end if

C       ******** reading data from disk *********
C
  222 continue
C
       read(lun,*,end=333) mseq, nstat,ncruise,
     * nyear,nmonth,nday,nhour,nmin,
     * gradlat,gradlon,ndepth,modepth,nobs,msq
      read(lun,*) MMAX
C
C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********
C-------BM Jahresangaben in *.dat sind von der Form "84" statt "1984"
        nyear = nyear + 1900
        leap_year = mod(nyear,4)

      if (((nhour.gt.24 .and. nhour.ne.99) .or. nhour.lt.0) .OR.
     *   (nday.gt.31 .or. nday.lt.1 ) .OR.
     *   (nmonth.gt.12 .or. nmonth.lt.1)  .OR.
     *   (nyear.gt.1994 .or. nyear.lt.1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute = '00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

        ELSE IF (nday .eq.29 .and.
        1        nmonth .eq. 2 .and.
        1        leap_year.ne.0) THEN

         Monat = 1
         Jahr = '1900'
         Tag = ' 1'
         Stunde ='00'
         Minute ='00'
        ELSE

         WRITE (TAG,'(I2)') nday
         WRITE (JAHR,'(I4)') nyear
         IF (nhour .eq. 24) THEN
          Stunde ='23'
         ELSE
          IF (nhour .eq. 99) THEN
              STUNDE = '00'
            ELSE
              WRITE (STUNDE,'(I2)') nhour
```

```
          END IF
         END IF
         IF (nmin .eq. 99) THEN
             Minute = '00'
           ELSE
             WRITE (Minute,'(I2)') nmin
         END IF
         MONAT=nmonth
        END IF

        ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
        2//STUNDE//':'//Minute//'"'

C       ***Speicherung der Stationsdaten***************

        ID_STAT=ID_STAT+1
        STATION.ID=ID_STAT
        STATION.CRUISE_NUMBER=ncruise
        STATION.STATION_NUMBER=nstat
        STATION.LATITUDE=gradlat
        STATION.LONGITUDE=gradlon
        STATION.BOTTOM_DEPTH=ndepth
        STATION.MAX_OBSE_DEPTH=modepth
        STATION.NUMBER_OBSE=nobs
        STATION.MARSDEN_SQUARE=msq

        type *, station.id,' ',ascii_time

        call fdbcmd(dbproc,' insert into BSH2_Station values ( ')
        call fdbfcmd(dbproc,' %d,', STATION.ID)
        call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
        call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
        call fdbfcmd(dbproc,' %s,', ASCII_TIME)
        call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
        call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
        call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
        call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
        call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

        call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

C       ********Speicherung der Messdaten*******

      do  k=1,MMAX
      read(lun,*)
     *zst(k),tst(k),sst(k),ost(k),sist(k),fnst(k),phst(k)

          DATA.DEPTH = zst(k)
          DATA.TEMPERATURE = tst(k)
          DATA.SALINITY = sst(k)
          DATA.OXYGEN = ost(k)
          DATA.PHOSPHATE = phst(k)
          DATA.SILICATE = sist(k)
          DATA.NITRAT = fnst(k)

        id_data=id_data+1
        DATA.ID=id_data
        DATA.STATION_ID = STATION.ID

        call fdbcmd(dbproc,' insert into BSH2_Data')
        call fdbfcmd(dbproc,' values (')
        call fdbfcmd(dbproc,' %d,', DATA.ID)
        call fdbfcmd(dbproc,' %d,', DATA.STATION_ID)
        call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
        call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
        call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
        call fdbfcmd(dbproc,' %f,', DATA.OXYGEN )
        call fdbfcmd(dbproc,' %f,', DATA.PHOSPHATE )
        call fdbfcmd(dbproc,' %f,', DATA.SILICATE )
        call fdbfcmd(dbproc,' %f)', DATA.NITRAT )
```

```
call fdbsqlexec(dbproc)
        return_code = fdbresults(dbproc)

        END DO

        GOTO 222

333   continue

      type *,'end of file'
      type*,'total of ',mseq
      close(lun)
      call fdbexit()
      END
```

for-27

REIDINTLOAD.FOR

```
options /check=all

C       CREATOR::M. REIKNE
C       CREA_DATE::25-Jul-1990
C       CHANGES:: 1994-05-06 BM
C                               reading Reid interpolated data
C
        structure /station/
        integer *4      ID
        integer *4      CRUISE_NUMBER
        integer *4      STATION_NUMBER
        real *8         LATITUDE
        real *8         LONGITUDE
        integer *4      BOTTOM_DEPTH
        integer *4      MAX_OBSE_DEPTH
        integer *4      NUMBER_OBSE
        integer *4      MARSDEN_SQUARE
        end structure

        structure /data/
        integer*4       ID
        integer*4       Station_ID
        real*8          TEMPERATURE
        real*8          SALINITY
        real*8          OXYGEN
        integer*4       DEPTH
        real*8          PHOSPHATE
        real*8          SILICATE
        real*8          NITRAT
        end structure

        record /STATION/ STATION
        record /DATA/ DATA

        include '(fsybdb)'
        include '($smgdef)'
        include '($ttdef)'
        include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                  err_handler
      EXTERNAL                  msg_handler

        INTEGER*4                 login,
        1                         dbproc,
        1                         return_code,
        1                         no_echo,
        1                         lun,
        1                         ipb,
        1                         id_stat,
        1                         id_data,
        1                         leap_year,
        1                         monat,
        1                         i

        character*4               Jahr
        character*2               Tag,
        1                         Stunde,
        1                         Minute
        character*3               month(12)

        character*30              ASCII_TIME

      INTEGER*4                 error
      CHARACTER*(256)           cmdbuf


      CHARACTER*20 password

C-------Victors Data and Formats
```

```
C
      character country10*10,ship18*18
C
      integer*4 nseq, codenodc

      real*8 phire, alamre

      integer*4 nyre, more, ndare, nhour, nmin, nstation, ncruise, nbdre,
     *          maxobsdre, msq, nobs, nst

      real*4 zst(42), fs2(42), fs3(42), fs4(42), fs5(42), fs6(42),
     *       fs7(42), fs8(42), fs9(42), fs10(42)
C

C       LIST OF VARIABLES
C_____________________________________________________________________
C CODENODC - NODC ship-country code (char*4)
C Phire - Latitude (grad.)
C Alamre - Longitude (grad.)
C nyre - Year
C more - Month
C Ndare - Day
C Nhour - Hour
C nmin  - Minutes
C Nstation - Originator's Station_Number
C Ncruise - SODB Cruise_Number
C Nbdre - Bottom_Depth (meters)
C Maxobsdre - Max_Obs_Depth (meters)
C Msq - Marsden Square
C Nobs - Number of observed levels
C Nst - Number of standard levels
C
C ZST(42) - array of standard depths (meters)
C fs2(42) -array of  Temperature (Grad C)
C fs3(42) - array of Salinity
C fs4(42) - array of Oxygen (ml/l)
C fs5(42) - array of Inorganic Phosphate (mkg.at/l)
C fs7(42) - array of Silicate (mkg.at/l)
C fs9(42) - array of Nitrate (mkg.at/l)
C


        DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
        2           'Sep','Oct','Nov','Dec'/


C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

C15    format(' Name of the input file: '$)
C 20    format(a50)
C      type 15
C      accept 20, file1
```

```
      call lib$get_lun(lun)
      open(unit=lun, file='oth$daten:[socean.reid]reid_sodb_int.dat',
     * status='old')


       call fdbfcmd(dbproc,
     1  'select max(Reid_Standard_Data_Id#) from Reid_Standard_Data')
       call fdbsqlexec(dbproc)
       call fdbresults(dbproc)
       call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
       call fdbnextrow(dbproc)

       if (ID_DATA .lt. 305000000) then
           ID_DATA = 305000000
       end if


c      ******** reading data from disk *********
C

501   format(2x, i10)

C--------- READ OBSERVED DATA
222        continue
           read(lun,501,end=333) nseq
           read(lun,300)codenodc
           read(lun,301)ship18,country10
           read(lun,*)  phire,alamre
           read(lun,*)  nyre,more,ndare,nhour,nmin
           read(lun,*)  nstation
           read(lun,*)  ncruise
           read(lun,*)  nbdre,maxobsdre,msq
           read(lun,*)  nobs
           read(lun,*)  nst


C      **Konstruktion des Zeitstrings
C      ***Testen ob Ausreisser in den Zeiten gibt ***********

       leap_year = mod(nyre,4)

       if (((Nhour.gt.24 .and. Nhour.ne.99) .or. Nhour .lt. 00) .OR.
       1   (Ndare.gt.31 .or. Ndare .lt. 1 ) .OR.
       1   (more.gt.12 .or. more .lt. 1)  .OR.
       1   (nyre.gt.1990 .or. nyre .lt. 1900)) then

       Monat = 1
       Jahr = '1900'
       Tag = ' 1'
       Stunde ='00'
       Minute = '00'

C      ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****

       ELSE IF (Ndare.eq.29 .and.
       1        more.eq. 2 .and.
       1        leap_year.ne.0) THEN

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute ='00'
       ELSE

        WRITE (TAG,'(I2)') Ndare
        WRITE (JAHR,'(I4)') nyre
        IF (Nhour .eq. 24) THEN
         Stunde ='23'
        ELSE
         IF (Nhour .eq. 99) THEN
             STUNDE = '00'
```

```
            ELSE
              WRITE (STUNDE,'(I2)') Nhour
           END IF
          END IF
          IF (nmin .eq. 99) THEN
              Minute = '00'
            ELSE
              WRITE (Minute,'(I2)') nmin
          END IF
          MONAT=more
         END IF

         ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
         2//STUNDE//':'//Minute//'"'

C-------Berechnen der Station.ID bei gegebener Originators Station ID
C
         call fdbfcmd(dbproc, 'select Reid_Station_Id# from Reid_Station')
         call fdbfcmd(dbproc, ' where Cruise_Number = %d', ncruise)
         call fdbfcmd(dbproc, ' and Station_Number = %d', nstation)
         call fdbfcmd(dbproc, ' and Date_Time = %s', ascii_time)
         call fdbfcmd(dbproc, ' and Longitude = %f', alamre)
         call fdbfcmd(dbproc, ' and Latitude = %f', phire)
         call fdbfcmd(dbproc, ' and Bottom_Depth = %d', nbdre)
         call fdbfcmd(dbproc, ' and Max_Obse_Depth = %d', maxobsdre)
         call fdbsqlexec(dbproc)
         call fdbresults(dbproc)
         call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
         if (fdbnextrow(dbproc) .eq. NO_MORE_ROWS) then

C----------neue Station in Reid_Station eintragen
         call fdbfcmd(dbproc,
     *               'select max(Reid_Station_Id#) from Reid_Station')
         call fdbsqlexec(dbproc)
         call fdbresults(dbproc)
         call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
         call fdbnextrow(dbproc)

C-------id_stat > 0, da reid_station nicht leer

         ID_STAT=ID_STAT+1
         STATION.ID=ID_STAT
         STATION.CRUISE_NUMBER=ncruise
         STATION.STATION_NUMBER=nstation
         STATION.LATITUDE=phire
         STATION.LONGITUDE=alamre
         STATION.BOTTOM_DEPTH=nbdre
         STATION.MAX_OBSE_DEPTH=maxobsdre
         STATION.NUMBER_OBSE=nobs
         STATION.MARSDEN_SQUARE=msq

         call fdbcmd(dbproc,' insert into Reid_Station values ( ')
         call fdbfcmd(dbproc,' %d,', STATION.ID)
         call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
         call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
         call fdbfcmd(dbproc,' %s,', ASCII_TIME)
         call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
         call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
         call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
         call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
         call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
         call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

         call fdbsqlexec(dbproc)
         return_code = fdbresults(dbproc)

         end if

C        ********Speicherung der Messdaten*******

         do k=1,nst
            read(lun,*)zst(k),fs2(k),fs3(k),fs4(k),fs5(k),fs7(k),fs9(k)
```

```
            DATA.DEPTH = zst(k)
            DATA.TEMPERATURE = fs2(k)
            DATA.SALINITY = fs3(k)
            DATA.OXYGEN = fs4(k)
            DATA.PHOSPHATE = fs5(k)
            DATA.SILICATE = fs7(k)
            DATA.NITRAT = fs9(k)

          id_data=id_data+1
          DATA.ID=id_data
          DATA.STATION_ID = ID_STAT

          call fdbcmd(dbproc,' insert into Reid_Standard_Data')
          call fdbfcmd(dbproc,' values (')
          call fdbfcmd(dbproc,' %d,', DATA.ID)
          call fdbfcmd(dbproc,' %d,', DATA.STATION_ID)
          call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
          call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
          call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
          call fdbfcmd(dbproc,' %f,', DATA.OXYGEN )
          call fdbfcmd(dbproc,' %f,', DATA.PHOSPHATE )
          call fdbfcmd(dbproc,' %f,', DATA.SILICATE )
          call fdbfcmd(dbproc,' %f)', DATA.NITRAT )
          call fdbsqlexec(dbproc)
          return_code = fdbresults(dbproc)

          END DO

          GOTO 222

300     format(2x,a4)
301     format(2x,a18,a10)

333       CONTINUE
          TYPE *,'end of file'
          TYPE *,' there are ',ID_STAT, ' stations in the file'
          CLOSE(LUN)
          call fdbexit()
          END
C       program r_reid_sodb_int
C
C READ interpolated REID'S DATA
C     V.Guretsky, AWI, Jan 94
```

for-28

```
options /check=all

C       CREATOR::M. REIKNE
C       CREA_DATE::25-Jul-1990
C       CHANGES:: 1994-05-04 BM
C                                reading Reid data
C
        structure /station/
        integer *4      ID
        integer *4      CRUISE_NUMBER
        integer *4      STATION_NUMBER
        real *8         LATITUDE
        real *8         LONGITUDE
        integer *4      BOTTOM_DEPTH
        integer *4      MAX_OBSE_DEPTH
        integer *4      NUMBER_OBSE
        integer *4      MARSDEN_SQUARE
        end structure

        structure /data/
        integer*4       ID
        integer*4       Station_ID
        real*8          TEMPERATURE
        real*8          SALINITY
        real*8          OXYGEN
        integer*4       DEPTH
        real*8          PHOSPHATE
        real*8          SILICATE
        real*8          NITRAT
        end structure

        record /STATION/ STATION
        record /DATA/ DATA

        include '(fsybdb)'
        include '($smgdef)'
        include '($ttdef)'
        include '($tt2def)'

C
C     Forward declarations of the error-handler and message-handler
C
      EXTERNAL                  err_handler
      EXTERNAL                  msg_handler

        INTEGER*4                 login,
        1                         dbproc,
        1                         return_code,
        1                         no_echo,
        1                         lun,
        1                         ipb,
        1                         id_stat,
        1                         id_data,
        1                         leap_year,
        1                         monat,
        1                         i

        character*4               Jahr
        character*2               Tag,
        1                         Stunde,
        1                         Minute
        character*3               month(12)

        character*30              ASCII_TIME

      INTEGER*4                 error
      CHARACTER*(256)           cmdbuf


      CHARACTER*20 password

C-------Victors Data and Formats
```

REIDOBSLOAD FOR

```
C
      character country10*10,ship18*18,codenodc*4
C
      integer*4 nseq

      real*8 phire, alamre

      integer*4 nyre, more, ndare, nhour, nmin, nstation, ncruise, nbdre,
     *          maxobsdre, msq, nobs

      real*4 z(900),f2(900),f3(900),f4(900),f5(900),f6(900),f7(900),
     *   f8(900),f9(900),f10(900)
C

C       LIST OF VARIABLES
C_____________________________________________________________________
C CODENODC - NODC ship-country code (char*4)
C Phire - Latitude (grad.)
C Alamre - Longitude (grad.)
C nyre - Year
C more - Month
C Ndare - Day
C Nhour - Hour
C nmin  - Minutes
C Nstation - Originator's Station_Number
C Ncruise - SODB Cruise_Number
C Nbdre - Bottom_Depth (meters)
C Maxobsdre - Max_Obs_Depth (meters)
C Msq - Marsden Square
C Nobs - Number of observed levels
C
C Z - array of standard depths (meters)
C f2 -array of  Temperature (Grad C)
C f3 - array of Salinity
C f4 - array of Oxygen (ml/l)
C f5 - array of Inorganic Phosphate (mkg.at/l)
C f7 - array of Silicate (mkg.at/l)
C f9 - array of Nitrate (mkg.at/l)
C
C INPUT FILE:
C       open(23,file='oth$daten:[socean.reid]reid_sodb_obs.dat',
C     * status='old')
C


       DATA MONTH /'Jan','Feb','Mar','Apr','May','Jun','Jul','Aug',
       2           'Sep','Oct','Nov','Dec'/

C
C     Install the user-supplied error-handling and message-handling
C     routines. They are defined at the bottom of this source file.
C
      call fdberrhandle(err_handler)
      call fdbmsghandle(msg_handler)
C
C     Allocate and initialize the LOGINREC record to be used
C     to open a connection to the DataServer.
C
      login = fdblogin()
      call fdbsetluser(login, 'sa')
      call ask_for_pw(password)
      call fdbsetlpwd(login, password)
C
C
C

C       *****Eroeffnen der Datenbank
C
      dbproc = fdbopen(login, NULL)
      call fdbuse(dbproc,'SouthernOceanDB')

C15    format(' Name of the input file: '$)
```

```
C 20    format(a50)
C      type 15
C      accept 20, file1
       call lib$get_lun(lun)
       open(unit=lun, file='oth$daten:[socean.reid]reid_sodb_obs.dat',
     * status='old')


        call fdbfcmd(dbproc,
        1       'select max(Reid_Station_Id#) from Reid_Station')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_STAT)
        call fdbnextrow(dbproc)

        if (ID_STAT .eq. 0) then
            ID_STAT = 3000000
        end if

        call fdbfcmd(dbproc,
        1  'select max(Reid_Data_Id#) from Reid_Data')
        call fdbsqlexec(dbproc)
        call fdbresults(dbproc)
        call fdbbind(dbproc,1,INTBIND,0,ID_DATA)
        call fdbnextrow(dbproc)

        if (ID_DATA .eq. 0) then
            ID_DATA = 300000000
        end if


c       ******** reading data from disk *********
C

501   format(2x, i10)
504   format(2x, 2f8.2)
505   format(2x, 5i7)
506   format(2x, i7)
508   format(2x, 3i4)
509   format(2x, i4)

C--------- READ OBSERVED DATA
222         continue
            read(lun,501,end=333) nseq
            read(lun,300)codenodc
            read(lun,301)ship18,country10
            read(lun,*)  phire,alamre
            read(lun,*)  nyre,more,ndare,nhour,nmin
            read(lun,*)  nstation
            read(lun,*)  ncruise
            read(lun,*)  nbdre,maxobsdre,msq
            read(lun,*)  nobs

C
C       **Konstruktion des Zeitstrings
C       ***Testen ob Ausreisser in den Zeiten gibt ***********

        leap_year = mod(nyre,4)

        if (((Nhour.gt.24 .and. Nhour.ne.99) .or. Nhour .lt. 00) .OR.
        1   (Ndare.gt.31 .or. Ndare .lt. 1 ) .OR.
        1   (more.gt.12 .or. more .lt. 1)  .OR.
        1   (nyre.gt.1990 .or. nyre .lt. 1900)) then

        Monat = 1
        Jahr = '1900'
        Tag = ' 1'
        Stunde ='00'
        Minute = '00'

C       ***Testen ob es in einem Nichtschaltjahr einen 29.2. gibt ****
```

```
ELSE IF (Ndare.eq.29 .and.
     1        more.eq. 2 .and.
     1        leap_year.ne.0) THEN

      Monat = 1
      Jahr = '1900'
      Tag = ' 1'
      Stunde ='00'
      Minute ='00'
     ELSE

      WRITE (TAG,'(I2)') Ndare
      WRITE (JAHR,'(I4)') nyre
      IF (Nhour .eq. 24) THEN
       Stunde ='23'
      ELSE
       IF (Nhour .eq. 99) THEN
           STUNDE = '00'
         ELSE
           WRITE (STUNDE,'(I2)') Nhour
       END IF
      END IF
      IF (nmin .eq. 99) THEN
          Minute = '00'
        ELSE
          WRITE (Minute,'(I2)') nmin
      END IF
      MONAT=more
     END IF

     ASCII_TIME='"'//MONTH(MONAT)//' '//TAG//' '//JAHR//' '
     2//STUNDE//':'//Minute//'"'

C    ***Speicherung der Stationsdaten***************

     ID_STAT=ID_STAT+1
     STATION.ID=ID_STAT
     STATION.CRUISE_NUMBER=ncruise
     STATION.STATION_NUMBER=nstation
     STATION.LATITUDE=phire
     STATION.LONGITUDE=alamre
     STATION.BOTTOM_DEPTH=nbdre
     STATION.MAX_OBSE_DEPTH=maxobsdre
     STATION.NUMBER_OBSE=nobs
     STATION.MARSDEN_SQUARE=msq

     type *, station.id,' ',ascii_time

     call fdbcmd(dbproc,' insert into Reid_Station values ( ')
     call fdbfcmd(dbproc,' %d,', STATION.ID)
     call fdbfcmd(dbproc,' %d,', STATION.CRUISE_NUMBER)
     call fdbfcmd(dbproc,' %d,', STATION.STATION_NUMBER)
     call fdbfcmd(dbproc,' %s,', ASCII_TIME)
     call fdbfcmd(dbproc,' %f,', STATION.LONGITUDE)
     call fdbfcmd(dbproc,' %f,', STATION.LATITUDE)
     call fdbfcmd(dbproc,' %d,', STATION.BOTTOM_DEPTH)
     call fdbfcmd(dbproc,' %d,', STATION.MAX_OBSE_DEPTH)
     call fdbfcmd(dbproc,' %d,', STATION.NUMBER_OBSE)
     call fdbfcmd(dbproc,' %d)', STATION.MARSDEN_SQUARE)

     call fdbsqlexec(dbproc)
     return_code = fdbresults(dbproc)

C    ********Speicherung der Messdaten*******

        do k=1,nobs
        read(lun,*)z(k),f2(k),f3(k),f4(k),f5(k),f7(k),f9(k)
        DATA.DEPTH = z(k)
        DATA.TEMPERATURE = f2(k)
        DATA.SALINITY = f3(k)
        DATA.OXYGEN = f4(k)
        DATA.PHOSPHATE = f5(k)
```

```
            DATA.SILICATE = f7(k)
            DATA.NITRAT = f9(k)

          id_data=id_data+1
          DATA.ID=id_data
          DATA.STATION_ID = STATION.ID

          call fdbcmd(dbproc,' insert into Reid_Data')
          call fdbfcmd(dbproc,' values (')
          call fdbfcmd(dbproc,' %d,', DATA.ID)
          call fdbfcmd(dbproc,' %d,', DATA.STATION_ID)
          call fdbfcmd(dbproc,' %d,', DATA.DEPTH)
          call fdbfcmd(dbproc,' %f,', DATA.TEMPERATURE)
          call fdbfcmd(dbproc,' %f,', DATA.SALINITY)
          call fdbfcmd(dbproc,' %f,', DATA.OXYGEN )
          call fdbfcmd(dbproc,' %f,', DATA.PHOSPHATE )
          call fdbfcmd(dbproc,' %f,', DATA.SILICATE )
          call fdbfcmd(dbproc,' %f)', DATA.NITRAT )
          call fdbsqlexec(dbproc)
          return_code = fdbresults(dbproc)

          END DO

          GOTO 222

300     format(2x,a4)
301     format(2x,a18,a10)

333       CONTINUE
          TYPE *,'end of file'
          TYPE *,' there are ',ID_STAT, ' stations in the file'
          CLOSE(LUN)
          call fdbexit()
          END
C       program r_reid_sodb_obs
C
C READ OBSERVED REID'S DATA
C      V.Guretsky, AWI, Jan 94
```

09.08.1991
Lkm

```
1> /*   AWI-Rechnergruppe */
2> /* created by lpk
3> ** 1991-08-09 */
4> **
5> ** FILE_NAME: MUENCHLOAD.SCRIPT
6> **
7> /* Dieses Script geht davon aus, dass die Datenbank SouthernOceanDB */
8> /* bereits existiert. */
9> use master
Msg 102, Level 15, State 1:
Server 'SYBASE401', Line 4:
Incorrect syntax near '*'.
1> /* Erzeugen der Datenbank */
2> /*   Die Datenbank muss nicht mehr erzeugt werden. */
3> if not exists (select * from master.dbo.sysdatabases
4>                   where name = "SouthernOceanDB")
5> begin
6> print 'SouthernOceanDB does not exist !!!'
7> return
8> end
9>
10> use SouthernOceanDB
(0 rows affected)
1>
2> /* Erzeugen der Tabellen */
3> if exists (select * from sysobjects where name="Muench_Station")
4>   drop table Muench_Station
1>
2>   create table Muench_Station
3>        (Muench_Station_Id#  int,      /* internal identification */
4>         Cruise_Number       int,      /* Muench CRUNU */
5>         Station_Number      int,      /* Muench NUMSTAT */
6>         Longitude           float,    /* Muench A */
7>         Latitude            float,    /* Muench P */
8>         Date_Time           datetime NULL,
9>                                       /* Muench nyear month nday nhour minut *♦
10>         Bottom_Depth        int NULL,/* Muench ndep */
11>         Max_Obse_Depth      int NULL,/* Muench modeph */
12>         Number_Obse         int,     /* Muench nlev */
13>         Marsden_Square#     int NULL /* Muench msq */)
1>
2> if exists (select * from sysobjects where name="Muench_Standard_Data")
3>   drop table Muench_Standard_Data
1>
2>   create table Muench_Standard_Data
3>        (Muench_Standard_Data_Id# int,         /* internal identification */
4>         Muench_Station_Id#       int,         /* internal identification */
5>         Depth                  int,         /* Muench ZZ */
6>         Temperature            float NULL,  /* Muench TEM */
7>         Salinity               float NULL  /* Muench SAL */)
8>
1>
2> /* Definition der Primaerschluessel in den Tabellen */
3>
4> execute sp_primarykey Muench_Station, Muench_Station_Id#
5> execute sp_primarykey Muench_Standard_Data, Muench_Standard_Data_Id#
New primary key added.
(return status = 0)
New primary key added.
(return status = 0)
1>
2> /* Definition der Sekundaerschluessel in den Tabellen */
3>
4> execute sp_foreignkey Muench_Standard_Data, Muench_Station,
5>         Muench_Station_Id#
New foreign key added.
```