

EpiData Software for Operations Research in Tuberculosis Control

A course developed in collaboration with the EpiData Association and the International Union Against Tuberculosis and Lung Disease



Authors:

Hans L **Rieder**, Kirchlintach, Switzerland

Jens M **Lauritsen**, Odense, Denmark

Major contributors to current content and recent revisions:

Nguyen Binh **Hoa**, Hanoi, Viet Nam

Ajay M V **Kumar**, Bengaluru, India

Zaw Myo Tun, Singapore

Date last revised: March 17, 2015

Note on versions of EpiData software: Always use the most recent release version of EpiData software which can be obtained freely from the EpiData website <http://www.epidata.dk>. The course is updated at least whenever a new version requires adaptation or following an in-class course, whichever comes earlier.

Course content

Part A: EpiData Entry

- Exercise 1 A data documentation sheet for a simple questionnaire
- Exercise 2 The QES-REC-CHK triplet
- Exercise 3 Derived fields and Check file commands unrelated to a specific field
- Exercise 4 Data entry and validation
- Exercise 5 Using an external file for label blocks
- Exercise 6 Dealing with incomplete dates
- Exercise 7 Keeping track of data entry time
- Exercise 8 Safely backing up and encrypting your data

Part B: EpiData Analysis

- Exercise 1 An introduction to EpiData Analysis
- Exercise 2 More on EpiData Analysis
- Exercise 3 Aggregating data and saving the summary data in a file

Part C: Operations research

- Exercise 1: Creating a working dataset
- Exercise 2: Variability in serial smears
- Exercise 3: Incremental yield from serial smears
- Exercise 4: Confirmatory results in serial smears

Part D: More on EpiData software

- Exercise 1: Relational database and aggregating vs from “Long-to-wide”
- Exercise 2: A statistical process control chart
- Exercise 3: A simplified survival analysis
- Exercise 4: Creating a menu for standard reports
- Exercise 5: Formatting standardized analysis output in a spreadsheet

Part E: Beyond EpiData Analysis using R

- Exercise 1: Introduction to R software: basics
- Exercise 2: Introduction to R software: data bases and functions
- Exercise 3: Multivariable analysis in R part 1: A logistic regression
- Exercise 4: Multivariable analysis in R part 2: A Cox proportional hazard model

Background, objective, and course history

Background

In the 1980ies, the United States Centers for Disease Control and Prevention (CDC) developed public health software for its Epidemic Intelligence Service. The software package was called Epi Info and, later supported and promoted by the World Health Organization, experienced an unprecedented spread throughout the world's public health community.

The initiative to make EpiData was taken by Jens M Lauritsen from Denmark. Initially it was conceived of as part of the "Initiative for Accident Prevention" at Funen County. Why was it necessary to develop a new data entry program, if Epi Info version 6 had all that was needed in terms of control of data entry and simplicity? With the development of the Windows[®] operating system the majority of current computer users found it increasingly harder to cope with the DOS[®] operating system of working in Epi Info. Furthermore, with the change in the operating system of computers to newer versions of the Windows[®] operating system, the Epi Info 6.04d based on the DOS[®] operating system was eventually becoming obsolete (Windows 7[®] is no more reading it). Nevertheless, the principles underlying Epi Info are fundamentally sound and needed to be preserved for the new generation of public health practitioners grown up with the Windows[®] operating system.

On the Epi Info discussion list there were some discussions on strategies around 1997-1998, when the Epi Info team at CDC in USA decided to make an updated Epi Info version 2000. The updated Epi Info applies a different strategy in using a completely new way of working and basing it on the Microsoft Access[®] database format instead of simple text files.

Commercially available proprietary software is not focused on documentation, simplicity of use and validation of double-entered data. EpiData Entry is a program with a focus on data entry. The ambition of EpiData Entry was to create a simple to use independent application, which would not interfere with or require any special database system drivers (dll-based routines) shared with or interfering with other applications.

EpiData software consists of two modules, EpiData Entry and EpiData Analysis. In this course you will be learning the use of both EpiData Entry (Part A) and EpiData Analysis (Part B), and to apply them to operations research (Part C), starting with its very basic functionality to increasing sophistication, and finally extend your programming skills in EpiData Analysis (Part D).

Course Objective:

Acquire the skills to capture research study data of high quality, supported by thorough documentation of every procedure, conduct basic analyses, and apply them to operations research.

History of the course

The data for the original course were collected in a study simultaneously conducted in Benin, Malawi, Nicaragua, and Senegal.

The persons who collaborated in study design, data collection, analysis, and writing of a report were Séverin **Anagonou** (Benin), Thuridur **Arnadottir** (The Union), Fatoumata **Ba** (Senegal), Awa Hélène **Diop** (Senegal), Donald A **Enarson** (The Union), Martin **Gninafon** (Benin), A C **Kasalika** (Malawi), Hans L **Rieder** (The Union), Tone **Ringdal** (The Union), Felix L M **Salaniponi** (Malawi), Alejandro A **Tardencilla Gutierrez** (Nicaragua) and Arnaud **Trébucq** (The Union).

Utilizing the data generated in this collaborative research project, **three courses** of two weeks duration each **from 1997 to 1999** with a total of 18 participants were conducted in Paris by The Union. Subsidies to defray costs of these three courses were borne by the United States Centers for Disease Control and Prevention, the Coopération Française, the Norwegian Heart and Lung Association, the International Organization for Migration, and the Korean Institute of Tuberculosis.

The **first course** was held in **April 1997** in Paris, France. The participants were Francis **Adatu-Engwau** (Uganda), Nora **Bonso-Bruce** (Ghana), Awa Hélène **Diop** (Senegal), Asma **Elsony** (Sudan), and Amina **Jindani** (The Union). The facilitators were Thuridur **Arnadottir**, Eric **Brenner** (USA), Lawrence J **Geiter** (The Union), Hans L **Rieder** (The Union), and Arnaud **Trébucq** (The Union).

The **second course** was held in **April 1998** in Paris, France. The participants were Mohammed **Akhtar** (Pakistan), Maurice **Andriamiandrisoa** (Madagascar), Manfred **Danilovits** (Estonia), Lew Woo Jin (Korea, Republic of), Nguyen Phuong **Hoa** (Vietnam), Shanta Bahadur **Pande** (Nepal), and Alejandro A **Tardencilla Gutierrez** (Nicaragua). The facilitators were Thuridur **Arnadottir** (The Union), Eric **Brenner** (USA), Lawrence J **Geiter** (The Union), Hans L **Rieder** (The Union), and Arnaud **Trébucq** (The Union).

The **third course** was held in **April 1999** in Paris, France. The participants were Ademir **de Albuquerque Gomes** (Brazil), Fatoumata **Ba** (Senegal), Ferdinand **Kassa** (Benin), Pushpa **Malla** (Nepal), **Tieng** Sivanna (Cambodia), and Wang Jie-Siu (China), and. The facilitators were Thuridur **Arnadottir** (The Union), Eric **Brenner** (USA), Lawrence J **Geiter** (The Union), Hans L **Rieder** (The Union), and Arnaud **Trébucq** (The Union).

Because of the large costs associated with conducting the course, the course content and format was tested through interactive distance learning during the year 2000 by email. We are particularly indebted to Panganai **Dhliwayo** (Zimbabwe) and Robert **Makombe** (Zimbabwe) who went through this course in their free time. This helped in clarifying ambiguities and removing errors and uploading the course in September 2000 to the Internet (<http://www.tbrieder.org>).

The **fourth course** was held in Addis Ababa, Ethiopia, in **April 2001**. The participants were Ahmed **Abdurehman** (Ethiopia), Jemal **Aliy** (Ethiopia), Mekdes Gebeyehu **Ayicheh** (Ethiopia), Abebe **Habte** (Ethiopia), Yohannes **Mengistu** (Ethiopia), Moustapha **Ndir** (Senegal), Serkalem **Tadesse** (Ethiopia), Betru **Tekle** (Ethiopia), Mohammed Ahmed **Yassin** (Ethiopia), and Getachew Eyob **Yitelelu** (Ethiopia). The facilitators were Nils E **Billo** (The Union), Panganai **Dhliwayo** (Zimbabwe), and Hans L **Rieder** (The Union).

In 2002, the course material was once more revised to replace Epi Info 6 with EpiData Entry for questionnaire design, data checks, data entry, and data validation. For data analysis the 6.04d DOS version of Epi Info was retained.

The **fifth course** was held in Paris, France, in **January 2003**. The participants were Nadia **Ait-Khaled** (The Union), Kya Jai Maug **Aung** (Bangladesh), **Chiang** Chen-Yuan (Taiwan), Paula I **Fujiwara** (The Union), Achilles **Katamba** (Uganda), **Kim** HeeJin (Korea, Republic of), Dumitru **Laticevschi** (Moldova), Jones **Michongwe** (Malawi) and Jotam G **Pasipanodya** (Zimbabwe). The facilitators were Panganai **Dhliwayo** (Zimbabwe), Robert **Makombe** (Zimbabwe), and Hans L **Rieder** (The Union).

The **sixth course** was held in Yangon, Myanmar, in **December 2003**. The participants were Nyein Nyein **Aye** (Myanmar), May Yee **Chan** (Myanmar), Tin Mi Mi **Khaing** (Myanmar), Thandar **Lwin** (Myanmar), Htar Htar **Oo** (Myanmar), Saw **Thein** (Myanmar), Ti **Ti** (Myanmar), Myo **Zaw** (Myanmar), and Thin Thin **Yee** (Myanmar). The facilitators were **Chiang** Chen-Yuan (The Union), Hans L **Rieder** (The Union), and I D **Rusen** (Canada). In all previous courses the hypothesis was to test whether The Union's assumption that ten suspects needed to be examined to identify one case of tuberculosis was applicable to the study area. During the course it emerged, however, that even refutation of the hypothesis had relatively little programmatic implication. It was thus decided to integrate the previously supplementary exercises (dealing with variability of positive findings, patterns of recorded results, and potential incremental yield) into one single hypothesis addressing the critical value of the number of smear examinations that may be justified to identify one additional case or failure on the third diagnostic or the second follow-up smear examination, respectively.

The **seventh course** was held in Paris, France, in **January 2004**. The participants were Wafaa Hassan **Ali Taha** (Sudan), Goar **Balasanants** (Russia), Nulda **Beyers** (South Africa), Kathy **Lawrence** (South Africa), Biggie **Mabaera** (Zimbabwe), Nymadawaa **Naranbat** (Mongolia), Mahshid **Nasehi** (Iran), Mauro **Occhi** (Mozambique), Yevgeniy **Shubin** (Russia), and Abigail **Wright** (World Health Organization). The facilitators were Panganai **Dhliwayo** (Zimbabwe), Jens M **Lauritsen** (EpiData Association, Denmark), Robert **Makombe** (Zimbabwe), and Hans L **Rieder** (The Union).

The **eighth course** was held in Berne, Switzerland, in **July 2004**. The participants were Sondhja **Bitter** (Switzerland), Eva **Blozik** (Switzerland), Lorenz **Borer** (Switzerland), Michael **Endrich** (Switzerland), Karin **Imoberdorf-Baumgartner** (Switzerland), Caroline **Keller** (Switzerland), Hansjörg **Lüthy** (Switzerland), Christoph **Pammer** (Austria), Sabine **Recker** (Switzerland), Kurt **Schmidlin** (Switzerland), Jan **Wagner** (Switzerland), Sabine **Walser** (Austria), and Mark **Witschi** (Benin). The facilitators were Hans L **Rieder** (The Union) and Marcel **Zwahlen** (University of Berne, Switzerland). The curriculum of this course was changed in two important aspects from the previous courses. First, a core curriculum was developed to allow the conduct of the course within five working days without loss of any relevant course aspects. Refinements of specific aspects were offered in addenda independent of each other. This allowed course completion within one week (core curriculum) or eight days (core and extended curriculum). Second, the Epi Info Analysis component was replaced by a beta test version of EpiData Analysis which proved to be working smoothly without any glitches.

The **ninth course** was held in Paris, France, in **January 2005**. The participants were Anneke **Hesseling** (South Africa), **Hu** Dongmei (China), Zanele **Hwalima** (Zimbabwe), **Liu** Zhentian (China), Henri **Luwaga** (Uganda), Nguyen Thien **Huong** (Vietnam), John **Osho** (Nigeria), Tran Ngoc **Buu** (Vietnam), Nevin **Wilson** (The Union), and Yao Hongyan (China). The facilitators were Panganai **Dhliwayo** (The Union), Jens M **Lauritsen** (EpiData Association, Denmark), Robert **Makombe** (Zimbabwe), and Hans L **Rieder** (The Union). In this course, only free public domain software was used. Data management and analysis were done with

EpiData and EpiData Analysis, supplemented where needed with spreadsheet functions of OpenOffice.

The **tenth course** was held in Berne, Switzerland, in **July 2005**. The participants were Thomas **Bart** (Switzerland), Lisanne **Christen** (Switzerland), Stephanie **Christensen** (Switzerland), Michael **Flück** (Switzerland), Yvonne **Jansen** (Switzerland), Irène **Marty** (Switzerland), Jeanne **Moser** (Switzerland), Christoph **Napierala** (Switzerland), Barbara **Prokup** (Germany), Martin **Raab** (Switzerland), Franziska **Rabenschlag-Trösch** (Switzerland), Claudia **Sauerborn** (Switzerland), and Yasemin **Yüksel** (Switzerland). The facilitators were Hans L **Rieder** (The Union) and Marcel **Zwahlen** (University of Berne, Switzerland). This course used the same software as the January 2005 course with the latest pre-release version of EpiData Analysis (Version 0.9, Release 5, Build 26). Input from the participants and faculty led to further streamlining of some exercises.

The **eleventh course** was conducted in Paris, in **January 2006**. The participants were Chay Sokun (Cambodia), Andrew D R C **Dimba** (Malawi), Elhafiz **Hussein Ibrahim** (Sudan), Akramul **Islam** (Bangladesh), Suksont **Jittimane** (Thailand), Le Van **Duc** (Vietnam), Nguyen Binh **Hoa** (Vietnam), Helmuth **Reuter** (South Africa), Ezra **Shimeles** (The Union), and Xu Min (China). The facilitators were Achilles **Katamba** (Uganda), Jens M **Lauritsen** (EpiData Association, Denmark), and Hans L **Rieder** (The Union). In September 2005, the EpiData Association released the stable EpiData Analysis Version 1.0. Between release and course commencement, the EpiData Association released upgrade Version 1.1 which added the last functionality (aggregate command) that had been possible previously only in Epi Info DOS Version 6.

During 2005 and 2006 the structure of the course was changed into three parts (EpiData Entry, EpiData Analysis, and Operations Research,). Parts A (EpiData Entry) and B (EpiData Analysis) were stripped of the operations research component for the benefit of those who do not have sufficient time available or who wish to familiarize themselves solely with the software. Conversely, Part C (Operations research) was stripped of components now introduced in Parts A and B, concentrating on the application of the latter to the example research project.

In **July 2006**, the **twelfth course** was conducted in Berne, Switzerland, for Master of Public Health students and other interested participants. These were Martin **Adam** (Switzerland), Edith **Betschart** (Switzerland), Tobias **Eckert** (Switzerland), Denise **Felber Dietrich** (Switzerland), T John **Kessler-Teuscher** (Switzerland), Esther **Kolb** (Switzerland), Brigitte **Kuenzle** (Switzerland), Sonia **Menéndez-González** (Switzerland), Stefan **Neuner-Jehle** (Switzerland), Christoph Paul **Röder** (Switzerland), and Hildebrand **Schwab** (Switzerland). The facilitators were Hans L **Rieder** (The Union) and Marcel **Zwahlen** (University of Berne, Switzerland).

In **August 2006**, the **thirteenth course** was given in Changsha, Hunan Province, China. All participants were from China. They were 范月玲 (**Fan Yueling**), 贾忠彬 (**Jia Zhongbin**), 阚晓宏 (**Kan Xiaohong**), 李晓凤 (**Li Xiaofeng**), 林岩 (**Lin Yan**), 邱柏红 (**Qiu Baihong**), 谭振 (**Tan Zhen**), 王铂 (**Wang Bo**), 汪清雅 (**Wang Qingya**), 张恩溥 (**Zhang Enpu**), 张宏伟 (**Zhang Hongwei**), 张会民 (**Zhang Huimin**), and 赵丁源 (**Zhao Dingyuan**). Facilitators were Chiang Chen-Yuan (The Union) and Hans L **Rieder** (The Union).

In December 2006, the **fourteenth** and **fifteenth courses** were given sequentially in Beijing, China. All participants were from China. In the thirteenth course the participants were 陈慧娟 (**Chen Huijuan**), 陈静 (**Chen Jing**), 陈伟 (**Chen Wei**), 房宏霞 (**Fang Hongxia**), 胡嘉 (**Hu Jia**), 梁路 (**Liang Lu**), 刘二勇 (**Liu Eryong**), 马斌忠 (**Ma Binzhong**), 王丹霞 (**Wang Danxia**), 吴顶峰 (**Wu**

Dingfeng), 于宝柱 (**Yu Baozhu**), and 张慧 (**Zhang Hui**). In the fourteenth course, the participants were 陈广华 (**Chen Guanghua**), 孔霞 (**Kong Xia**), 李曙光 (**Li Shuguang**), 罗丹 (**Luo Dan**), 马艳 (**Ma Yan**), 庞学文 (**Pang Xuewen**), 司马雅云 (**Sima Yayun**), 徐吉英 (**Xu Jiyin**), 依帕尔 (**Yi Pa'er**), 张广恩 (**Zhang Guang'en**), 赵津 (**Zhao Jin**), and 周扬 (**Zhou Yang**). Facilitators were Jens M **Lauritsen** (EpiData Association, Denmark), Biggie **Mabaera** (University of Zimbabwe, Zimbabwe), and Hans L **Rieder** (The Union), with the assistance of 胡冬梅 (**Hu Dongmei**), 徐敏 (**Xu Min**), and 张慧 (**Zhang Hui**).

In **June 2007**, the **sixteenth course** was given in Khartoum, Sudan. Two major changes were made to the course curriculum. The first change was that the database for Part C was changed to the utilization of the dataset collected as a course project of the January 2003 and 2004 courses (data courtesy: Dumitru **Laticevschi**, Moldova; Nymadawaa **Naranbat**, Mongolia; Achilles **Katamba**, Uganda; Biggie **Mabaera**, Zimbabwe). The second change was to use the test version 2.0 of EpiData Analysis. All participants were from Sudan. The participants were خديجة ادم محمد (Khadiga **Adam** Mohammed), اسرار محمد عبدالسلام (Asrar M A/Salam **Elegail**), معاذ سرالختم اسماعيل (Maaz Sier Elkhatim Ismail), حباب خالد الخير (Habab Khalid **Elkheir** Omer), امل السمانى اسماعيل (Amel Elsammani Mohamed Ahmed **Elmuozamil**), مناضل حسن محمد علي (Monadil **Hassan** Mohamed Ali), سيد محمد همت (Sayed Mohammed Shareef **Himat**), ثويبة عمر علي محقر (Thoeiba Omer Ali **Muhagger**), عبدالمجيد عثمان موسى (Abdelmageed Osman **Musa**), علا محمود رحمة الله (Olla Mahmoud **Rahamtalla**) and عزمي عبدالرحمن عمارة (Azmi **Omara**). The coordinator was لوران علي زين العابدين (Louran **Zein Abdin Ali**, National Tuberculosis Programme Sudan), and facilitators were الحافظ حسين ابراهيم (Elhafiz Hussein **Ibrahim**, Epi-Lab, Sudan), and Hans L Rieder (The Union).

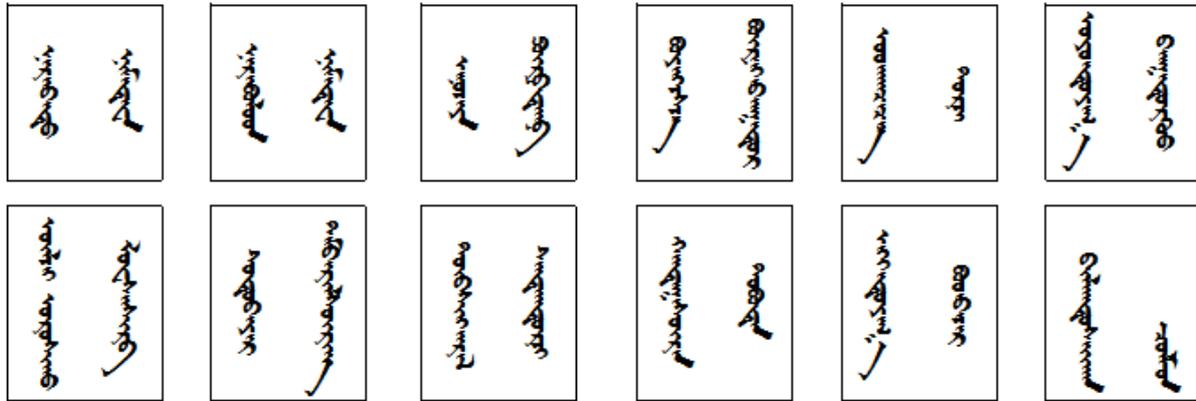
In **July 2007**, the **seventeenth course** was given in Berne, Switzerland, for Master of Public Health students and other interested participants. These were Nicole **Bender-Oser** (Switzerland), Bettina **Bringolf-Isler** (Switzerland), Sabina **Büttner** (Switzerland), Christian **Frei** (Switzerland), Florian **Gutzwiller** (Switzerland), Peter **Heri** (Switzerland), Kerstin **Hug** (Switzerland), André B **Kind** (Switzerland), Dimitri **Korol** (Switzerland), Cornelia **Marti** (Switzerland), Anne **Spaar** (Switzerland), and Françoise **Teuscher** (Switzerland). The facilitators were Hans L **Rieder** (The Union) and Marcel **Zwahlen** (University of Berne, Switzerland).

In **November 2007**, the course material was updated to make minor changes to Part A and to accommodate the changes introduced with the release Version 2.0 of EpiData Analysis.

In **April 2008**, the **eighteenth course** was given in Chisinau, Moldova. EpiData Version 3.1 (Build 270108) and EpiData Analysis Version 2.0 (Pre-release 2.1 Test Build 132) was utilized. The participants were Əlixanova Natəvan (**Alikhanova** Natavan), Azerbaijan; uCa nanava (Ucha **Nanava**), Georgia; maia qavTaraZe (Maia **Kavtaradze**), Georgia; Otilia **Scutelnicu**, Moldova; Angela **Capcelea**, Moldova; Rita **Seicas**, Moldova; Viorel **Soltan** Moldova; Ştefan **Savin**, Moldova; nino lomTaZe (Nino **Lomtadze**), Georgia; Constantin Dan **Nicolaiciu**, Romania; Iuliana **Husar**, Romania; Richard **Oleko** (Sudan). Facilitators were Jens M **Lauritsen** (EpiData Association, Denmark), Biggie **Mabaera** (Temporary Consultant to The Union, Lesotho), and Hans L **Rieder** (The Union).

In **June 2008**, the **nineteenth course** was given in Ulaanbaatar, Mongolia. EpiData Version 3.1 (Build 270108) and EpiData Analysis Version 2.0 (Pre-release 2.1 Test Build 132) were utilized. The participants were Пүрэвдагва Анузаяа (**Anuzaya** Purevdagva), Цолмон Билэгтсайхан (**Bilegtsaikhan** Tsolmon), Бүрнээбаатар Буянхишиг (Burneebaatar **Buyankhishig**), Бүдбазар Энхтуяа (Budbazar **Enkhtuya**), Довдон Хандаасүрэн (Dovdon **Khandaasuren**), Н Наранболд (Н. Наранболд), Ваатархуу Оюунтуяа (Баатархүү Оюунтуяа), Dorj **Otgontsetseg** (Дорж Отгонцэцэг), Demberelsuren **Sodbayar**

(Дэмбэрэлсүрэн Содбаяр), Sandagdorj **Tuvshingerel** (Сандагдорж Түвшингэрэл), Luvsansharav **Ulzii-Orshikh** (Лувсаншарав Өлзий-Орших). Hans L Rieder (The Union) was the facilitator. Following are the names of Nymadawa **Naranbat**, who organized the course, and the eleven participants written in the traditional Mongolian alphabet.



In June / July 2008, the **twentieth** course was given in Berne, Switzerland, for Master of Public Health students and other interested participants. EpiData Version 3.1 (Build 270108) and EpiData Analysis Version 2.0 (Pre-release 2.1 Test Build 132) were utilized. All participants were from Switzerland. They were Caroline **Bähler-Baumgartner**, Daniela **Dyntar**, Martin **Egger**, Carola A **Huber**, Ursula **Kälin-Keller**, Annette **Koller Doser**, Andrea **Merkel-Hoek**, Eric **Odenheimer**, Helen **Prytherch**, Anna **Späth**, and Melinda **Spiesshofer**. The facilitators were Hans L **Rieder** (The Union) and Marcel **Zwahlen** (University of Berne, Switzerland).

In **October 2008**, the **twenty-first** course was given in Kampala, Uganda. Course I was given during this five-day course. It was specifically designed to fit this 5-day course, also incorporating the previously existing brief overview of EpiData software. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.0 (Pre-release 2.1 Test Build 141) were utilized. The participants were Raymond **Asiimwe** (Uganda), Bernard Ssentalo **Bagaya** (Uganda), Freddy **Bwanga** (Uganda), Henry **Byabajungu** (Uganda), Basra Esmail **Doulla** (Tanzania), Nicholas **Ezati** (Uganda), Fred **Kangave** (Uganda), George **Lukyamuzi** (Uganda), Diana **Nadunga** (Uganda), and Raymond **Shirima** (Tanzania). Francis **Adatu-Engwau** (Uganda) was a guest participant, and formerly a participant in the first course. The facilitator was Hans L **Rieder** (The Union), supported in part by Achilles **Katamba** (Makerere University, Uganda, and The Union Country Office, Uganda).

In **February 2009**, the **twenty-second** course was given in Kampala, Uganda. Course II, Part B (EpiData Analysis) and Part C (Operations research) were given during this five-day course, although some exercises had to be skipped due to the brief duration of the course. EpiData Analysis Version 2.1 (Test Build 159) was utilized. The participants were Francis **Adatu-Engwau** (Uganda), Bernard Ssentalo **Bagaya** (Uganda), Freddy **Bwanga** (Uganda), Basra Esmail **Doulla** (Tanzania), Fred **Kangave** (Uganda), Diana **Nadunga** (Uganda), and Raymond **Shirima** (Tanzania). The facilitator was Hans L **Rieder** (The Union), supported in part by Achilles **Katamba** (Makerere University, Uganda, and The Union Country Office, Uganda).

In **July 2009**, the **twenty-third** course was given in Berne, Switzerland, for Master of Public Health students and other interested participants. EpiData Version 3.1 (Build 270108) and EpiData Analysis Version 2.2 (Build 169) were utilized. All but one participant from the principality of Liechtenstein were residents of Switzerland. They were Aline **Barbir**, Rita

Born, Mazda Farshad, Oliver Fuchs, Juliette Gerber, Gerti Kitting Gaillard, Meltem Kutlar Joss, Teresa Leisebach Minder, Elisabeth Oberfeld, Anna Plym, Corinna Rüegg, Dino Schlamp, Eliane Siegenthaler, Federico Soldati, and Esther Walser (Principality of Liechtenstein). The facilitators were Hans L **Rieder** (The Union) and Marcel **Zwahlen** (University of Berne, Switzerland).

In **October 2009**, the **twenty-fourth** course was given in Paris, France, for fellows and participants in the training module 2 offered by the Centre for Operational Research of The Union. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.171 were utilized. The participants were Dawit **Assefa Lemma** (Ethiopia), Walter Kizito **Kibango** (Kenya), Proscovia Namuwenge **Mukonzo** (Uganda), Mweete Debra **Nglazi** (South Africa), **Nguyen Binh Hoa** (Viet Nam), Mahfuza **Rifat** (Bangladesh), Srinath **Satyanarayana** (India), Zaw Myo **Tun** (Myanmar), Hannock Mukoma **Tweya** (Malawi) and Susanna Sophia **Van Wyk** (South Africa). The facilitators were Hans L **Rieder** (The Union) and Anthony D **Harries** (The Union), assisted by Nguyen Binh **Hoa** (Fellow, Viet Nam).

In **March 2010**, the **twenty-fifth** course was given in Paris, France, for three selected fellows and participants in the training module 2 offered by the Centre for Operational Research of The Union. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.171 were utilized. The course focused on Parts C (Operations Research) and D (More on EpiData software). The participants Nguyen Binh **Hoa** (Viet Nam), Zaw Myo **Tun** (Myanmar), and Hannock Mukoma **Tweya** (Malawi). The facilitators were Hans L **Rieder** (The Union) and Jens M **Laurtisen** (EpiData Association).

In **July 2010**, the **twenty-sixth** course was given in Berne, Switzerland, for Master of Public Health students and other interested participants. EpiData Version 3.1 (Build 270108) and EpiData Analysis Version 2.2 (Build 171) were utilized. All participants were residents of Switzerland. They were Brigitte **Brunner**, Adrian **Businger**, Elisabeth **Maurer Schild**, Rebecca **Osterwalder**, Alexandra **Rauch**, Charles **Senessie**, Ulf **Tölle**, and Roco **Umbescheidt**. The facilitators were Hans L **Rieder** (The Union) and Marcel **Zwahlen** (University of Berne, Switzerland).

In **October 2010**, the **twenty-seventh** course was given in Paris, France, for fellows and participants in the training module 2 offered by the Centre for Operational Research of The Union. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.171 were utilized. The participants were Felix **Afutu** (Ghana), Sarabjit Singh **Chadha** (India), Karen **Du Preez** (South Africa), Razia **Fatima** (Pakistan), Oliver **Gadabu** (Malawi), Lucy **Guluka-Gawa** (Malawi), Mohammed **Kogali** (Sudan), Rose Jepchumba **Kosgei** (Kenya), Ajay M V **Kumar** (India), Tonderayi Clive **Murimwa** (Zimbabwe), Mauro **Niskier Sanchez** (Brazil), and Kudakwashe Collin **Takarinda** (Zimbabwe). The facilitators were Hans L **Rieder** (The Union), Nguyen Binh **Hoa** (Viet Nam), Zaw Myo **Tun** (Myanmar), and Sven Gudmund **Hinderaker** (The Union).

In **May 2011**, the **twenty-eighth** course was given in Paris, France, for two selected fellows from the training module 2 offered by the Centre for Operational Research of The Union. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.171 were utilized. The course focused on Parts C (Operations Research) and D (More on EpiData software). The participants Karen **Du Preez** (South Africa) and Ajay M V **Kumar** (India). The facilitators were Hans L **Rieder** (The Union), Jens M **Laurtisen** (EpiData Association), and Hannock Mukoma **Tweya** (Malawi).

In **November 2011**, the **twenty-ninth** course was given in Paris, France for fellows and participants in the training module 2 offered by the Centre for Operational Research of The

Union. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.171 were utilized. The participants were Henry Shadreck **Kanyerere** (Malawi), Nicholas **Kirui** (Kenya), Pranay **Lal** (India), Sharath Bugurina **Nagaraja** (India), Sharan **Ram** (Fiji), Carlos Alberto **Mendoza-Ticono** (Peru), Emmanuel **Singogo** (Malawi), Nelda **van Soelen** (South Africa), Kerry **Viney** (New Caledonia), and Aung Naing **Win** (Myanmar). The facilitators were Hans L **Rieder** (The Union), Sarabjit **Chadha** (The Union), and Ajay M V **Kumar** (India).

In **February 2012**, the **thirtieth** course was given in Kathmandu, Nepal for fellows and participants in the training module 1b offered by the Centre for Operational Research of South East Asia office of The Union. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.178 were utilized. The participants were Tshering **Jamtsho** (Bhutan), Tashi **Denup** (Bhutan), Shyla **Islam** (Bangladesh), Wasantha **Jayakodi** (Sri Lanka), Swati **Srivastava** (India), Suresh **Shastri** (India), Ramya **Ananthakrishna** (India), Rajendra **Basnet** (Nepal), Caetano **Gusmao** (Timor Leste), Sokhan **Khann** (Cambodia), Sarwat **Shah** (Pakistan), Iwayan **Artwan** (Indonesia). The facilitators were Ajay M V **Kumar** (The Union, India) and Srinath **Satyanarayana** (The Union, India).

In **March 2012**, the **thirty-first** course was given in Bengaluru, India, for participants of the operational research course conducted by the Centre for Operational Research of South East Asia office of The Union in association with the National Tuberculosis Programme, United States Centre for Disease Control and Prevention, and the National Tuberculosis Institute. EpiData Entry Version 3.1 (Build 270108) was utilized. The participants (all residing in India and working for or in close association with the national tuberculosis programme) were Gurpeet **Singh**, Priyanka **Agrawal**, **Subhakar**, Raju **Chepuri**, Solomon **Muller Ankala**, Chetan **Purad**, D J **Deka**, Rajeev **Pathak**, Pankaj **Nimavat**, Amar **Shah**, Sunil **Kumar**, Karthickeyan **DSA**, Rajesh **Deshmukh**, Tapas **Patra**, Parija **Debudatta**, Asha **Fedrick**, Shivramakrishnan, Rajabhau D **Yeole**, Suparna **Chatopadhaya**, Shiv Kumar **Singh**, Vrinda **Sahasrabhojane**, **Anand** and **Tripathi**. The facilitators were Ajay M V **Kumar** (The Union, India) and Srinath **Satyanarayana** (The Union, India).

In **April 2012**, the **thirty-second** course was given in Paris, France, for three selected fellows from the training module 2 offered by the Centre for Operational Research of The Union. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.178 were utilized. The course focused on Parts C (Operations Research) and D (More on EpiData software). The participants were Sarabjit Singh **Chadha** (India), Sharath **Burugina Nagaraja** (India), and Nicholas **Kirui** (Kenya). The facilitators were Hans L **Rieder** (The Union), Ajay M V **Kumar** (The Union, India), and Jens M **Laurtisen** (EpiData Association).

In **May 2012**, the **thirty-third** course was conducted in Goa, India, for about 60 or more WHO-hired consultants working for the national tuberculosis programme in India (plus a couple of observers from the WHO-India and WHO-SEARO office) to build their capacity in data management; they were in turn expected to train all the 650 and more data entry operators working for the national tuberculosis programme. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.178 were utilized. The course duration was three days and focused on Parts A (Data Entry) and B (Data Analysis). The facilitators were Ajay M V **Kumar** (The Union, India), Sharath **Burugina Nagaraja** (India), Kiran **Rade** (India) and Srinath **Satyanarayana** (The Union, India).

In **May-June 2012**, the **thirty-fourth** course was given in Nairobi, Kenya, as a component of a course on data management for personnel working in tuberculosis reference laboratories, largely in Africa. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version

2.2.1.178 were utilized. The course focused on Parts A (Data Entry) and B (Data Analysis). The participants were Pamela Juma **Akinyi** (Kenya), Vignon Charles Frank **Faihun** (Bénin), Frederick **Kangave** (Uganda), Thuwein Yusuf **Makamba** (Tanzania), Faisal **Masood** (Pakistan), Margaret **Ndisha** (Kenya), Diana **Nadunga** (Uganda), Collins Tanaka **Sakubani** (Zimbabwe), Tigist Habtamu **Shiferaw** (Ethiopia), Elizabeth Ndaafetwa **Shipiki** (Namibia), Raymond Philip **Shirima** (Tanzania), and Mukururi **Sibanda** (Botswana). The facilitators were Hans L **Rieder** (The Union) and Armand **Van Deun** (The Union). Subsequent to the course, the structure of Part A (EpiData Entry) was adjusted to improve the flow into more coherent blocks.

In **July 2012**, the **thirty-fifth** course was given in Berne, Switzerland, for Master of Public Health students and other interested participants. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2 (Build 178) were utilized. The participants (all residing in Switzerland) were Lukas **Fenner**, Micòl **Gianinazzi**, Lotte **Habermann-Horstmeier**, Eva-Maria **Hau-Grosch**, Marianne **Jost**, Tanja **Löhri**, Thomas **Plattner**, Regina **Scharf**, Laura **Wengenroth**, Alfred **Wiesbauer**, and Natascha M **Wyss**. The facilitators were Hans L **Rieder** (The Union) and Marcel **Zwahlen** (University of Berne, Switzerland).

In **July 2012**, the **thirty-sixth** course was given in Paris, France, for fellows and participants in a training module offered by the Centre for Operational Research of The Union. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.178 were utilized. The participants were Serge **Ade** (Benin), Fengling **Mi** (China), Emmanuel **Kanike** (Malawi), Bodrun **Siddiquea** (Bangladesh), Kang Yang **Lim** (Singapore), Prashant **Bhat** (India), Ganzaya **Sukhbaatar** (Mongolia), Hilary **Gunguwo** (Zimbabwe), Philip **Owiti** (Kenya), Eunice **Wahome** (Kenya), Christine **Njuguna** (South Africa) and Kesete **Yirdaw** (Ethiopia). The facilitators were Ajay M V **Kumar** (The Union, India), Nguyen Binh **Hoa** (Viet Nam) and Srinath **Satyanarayana** (The Union, India).

In **September 2012**, the **thirty-seventh** course was given in Chennai, India, for participants of the operational research course conducted by the South East Asia office of The Union in association with the National Institute for Research in Tuberculosis (erstwhile Tuberculosis Research Centre). EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.178 were utilized. The participants (all residing in and working for tuberculosis control in India) were Kovid **Sharma**, Jaya Prasad **Tripathy**, Jaiswal **Narendra**, Jigneswar **Patel**, Nanda **Kumar**, Sanat Kumar **Tripathi**, Kamal Chand **Naik**, Lord Wasim **Reza**, Priyakanta **Nayak**, Ramesh **Kumar**, **Palanivel C** and **Deepa D**. The facilitators were Ajay MV **Kumar** (The Union, India), Ramya **Ananthkrishnan** (REACH, India) and Amar **Shah** (WHO, India).

In **December 2012**, the **thirty-eighth** course was given in Nadi, Fiji Islands, for participants of the Pacific Operational Research Course conducted by the Centre for Operational Research of The Union and The Secretariat of the Pacific Community. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.1.178 were utilized. The participants, from different countries of the Pacific region were Edwin Henry **Daiwo** (Solomon Islands), Rupihner **Defang** (Federated States of Micronesia), Saen **Fanai** (Vanuatu), Louise **Fonua** (Tonga), Natalie **Girin** (New Caledonia), Mareta **Hauma** (Republic of Marshall Islands), Noel **Itogo** (Solomon Islands), Tom **Jack** (Republic of Marshall Islands), Veisia **Matoto** (Tonga), Joaquin **Nasa** (Republic of Marshall Islands), Markleen **Tagaro** (Vanuatu) and Karen **Tairea** (Cook Islands). The facilitators were Ajay M V **Kumar** (The Union, South East Asia office and also module chair), Karen **Bissell** (The Union), Shakti **Gounder** (Fiji Ministry of Health), Bridget **Kool** (The University of Auckland, New Zealand), Sharan **Ram**

(Fiji National University), Christine **Roseveare** (EpiData Trainer, New Zealand) and Kerri Viney (Secretariat of the Pacific Community).

In **January 2013**, the **thirty-ninth** course on “Efficient, Quality assured Data capture using EpiData” was given in Bengaluru, India, for medical college professionals working in close collaboration with the Revised National TB Control Programme in Karnataka. This course was conducted by the South-East Asia office of The Union in association with the Bangalore Medical College and Research Institute. This course, the first of its kind in India, utilized EpiData Entry Version 3.1 (Build 270108). The participants were **Madhusudan M**, Mohammed **Imran**, Amit Kumar **Rao**, Arshiya **Kouser S**, Navya **CJ**, **Vinay M**, Nagaraja **Goud B**, **Ranganath TS**, **Ravish KS**, **Kishore SG**, **Sarsawathi S**, **Thilak SA**, **Shivaraj BM**, Riyaz **Basha S**, Nimmy **Dominic**, Ashok Kumar **Gudagunti**, Deepak **Tamang**, Surbhi **Joshi**, Karuna D **Sagili**, Archana **Trivedi**, Swaroop Kumar **Sahu**, Josephine **Priya K**, Suresh **Kumbhar**, **Anushka T**, Lilian **Dsouza**, Nevin **Wilson** and Sunita **Prasad**. The facilitators were Ajay M V **Kumar** (The Union, South East Asia office and also module chair), Balaji **Naik** (WHO-RNTCP, Bengaluru), **Deepak KG** (WHO-RNTCP, Tumkur), Prashant **Bhat** (WHO-RNTCP, Hubli) and **Palanivel** (IGMCRI, Puducherry).

In **February-March 2013**, the **fortieth** course was given in Kathmandu, Nepal for participants of the Asian Operational Research Course conducted by the Centre for Operational Research of South-East Asia office of International Union Against Tuberculosis and Lung Disease (The Union). EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.2.180 were utilized. The participants were Mangal **Bahadur Tharu** (Nepal), Xia **Yinyin** (China), Yaping **Chang** (China), Lekey **Khandu** (Bhutan), Jakir Hossain **Bhuiyan Masud** (Bangladesh), Sumudu Chandana **Abeygunawardena** (Srilanka), Olga **Denisiuk** (Ukraine), Gabeena **Mamoon** (Pakistan), Satyavani (India), Mrinalini **Das** (India), Shankar **D** (India), Ravinder **Kumar** (India). The facilitators were Ajay M V **Kumar** (The Union, India), Swati **Srivastava** (PHFI, India), Balaji **Naik** (WHO-RNTCP, India), Karuna D **Sagili** (The Union, India) and Deepak **Tamang** (The Union, India).

In **March 2013**, the **forty-first** course on “Efficient, Quality-assured Data capture using EpiData” was given for medical college professionals and scientists from the Vector Control Research Centre working in Puducherry, India. This course was conducted by the South-East Asia office of The Union in association with the medical college Jawaharlal Institute of Postgraduate Medical Education and Research (JIPMER). This course utilized EpiData Entry Version 3.1 (Build 270108). The participants were **Murugan V**, Prasad **Dikale**, Kavita **Vasudevan**, Hemant **Shewade**, Johnson **Cherian**, Mahalakshmi, **Subitha L**, Anindo **Majumdar**, **Kalaiselvi**, **Vedapriya**, Anuj **Mittal**, Ramesh **Chauhan**, Madhan **Raj**, S **Srikanth**, E Susiganesh **Kumar**, R **Kanagarethinem**, R **Moorthy**, Yogesh A **Bahurupi**, Om Prakash **Bera**, Surendar **Rangaswamy**, Manoj Kumar **Panigrahi**, R **Srinivasan**, C **Sadanandane** and **Ganesh**. The facilitators were Ajay M V **Kumar** (The Union, South-East Asia office and also module chair), **Palanivel** (IGMCRI, Puducherry), Swaroop Kumar **Sahu** (JIPMER, Puducherry), Karuna D **Sagili** (The Union, India) and Deepak **Tamang** (The Union, India).

In **March 2013**, the **forty-second** course was given in Bengaluru, India, for participants of the operational research course conducted by the Centre for Operational Research of South-East Asia office of The Union in association with the India National Tuberculosis Programme, United States Centers for Disease Control and Prevention (CDC Atlanta), and the National Tuberculosis Institute, Bangalore. EpiData Analysis Version 2.2.2.180 was utilized. The participants (all residing in India and working for or in close association with the national tuberculosis programme) were Gurpeet **Singh**, Priyanka **Agrawal**, Raju **Chepuri**, D J **Deka**,

Rajeev **Pathak**, Amar **Shah**, Sunil **Kumar**, **Karthickeyan** DSA, Rajesh **Deshmukh**, Parija **Debudatta**, Asha **Fedrick**, **Shivramakrishnan**, Rajabhau D **Yeole**, **Anand** and **Tripathi**. The facilitators were Ajay M V **Kumar** (The Union, India), Srinath **Satyanarayana** (The Union, India), Balaji **Naik** (WHO-RNTCP, India), Sreenivas **Nair** (WHO-India), Vineet K **Chadha** (NTI, India), Patrick **Moonan** (CDC Atlanta), Smitha **Ghosh** (CDC Atlanta) and John **Oeltmann** (CDC Atlanta).

In **March 2013**, the **forty-third** course on “Efficient, Quality-assured Data Analysis using EpiData” was given in Bengaluru, India, for medical college professionals working in and around Bengaluru, Karnataka. This course was organized by Bangalore Medical College and Research Institute in collaboration with the South-East Asia office of The Union. This course utilized EpiData Analysis Version 2.2.2.180. The participants were Amit Kumar **Rao**, **Ashwini**, Navya **CJ**, Nagaraja **Goud B**, **Ranganath TS**, **Ravish KS**, **Sarsawathi S**, **Thilak SA**, **Shivaraj BM**, Riyaz **Basha S**, Nimmy **Dominic**, Swaroop Kumar **Sahu**, Josephine **Priya K**, Lilian **Dsouza**, **Hamsa L**, Puneeth **Kumar** and Subhash **Babu P**. The facilitators were Ajay M V **Kumar** (The Union, South-East Asia office and also module chair), Karuna D **Sagili** (The Union, India) and Deepak **Tamang** (The Union, India).

In **April 2013**, the **forty-fourth** course on “Advanced use of EpiData software for operations research” was conducted in Delhi, India, with an emphasis on Parts B, C, and D, using EpiData Entry Version 3.1 (270108) and EpiData Analysis Version 2.2.2.180. The participants, all from India, were Jaya Prasad **Tripathy**, **Palanivel C**, Amar N **Shah**, Balaji **Naik**, Priyakanta **Nayak**, Debadutta **Parija**, Mrinalini **Das**, Deepak **Tamang**, Karuna Devi **Sagili**, and Riyaz **Basha Sardar**. The facilitators were Ajay M V **Kumar** (The Union, South-East Asia office) and Hans L **Rieder** (The Union, Paris).

In **May 2013**, the **forty-fifth** course was conducted in Yaoundé, Cameroun, using EpiData Entry Version 3.1 (270108) and EpiData Analysis Version 2.2.2.181. The participants were Wilfried **Békou Kossi** (Bénin), Frédéric **Békang Angui** (Cameroun), André **Nana Yakam** (Cameroun), Yvon Martial **Ngana** (Centrafrique), Jules **Toloko Risasi** (Congo, RD), Valéri **Oulaï Ibodé** (Côte d’Ivoire). The facilitator was Hans L **Rieder** (The Union, Paris).

In **May-June 2013**, the **forty-sixth** course was conducted in Paris, France, using EpiData Entry Version 3.1 (270108) and EpiData Analysis Version 2.2.2.182. The course title was “Advanced use of EpiData software for operations research” and was given to four participants selected from the Union’s Centre for Operational Research 2012 cohort who had already passed “Module 1b”. The course focused on Parts B (Introduction to EpiData Analysis), C (Operations Research) and D (More on EpiData software). The participants were Kesete **Yirdaw** (Ethiopia), Prashant **Bhat** (India), Philip **Owiti** (Kenya), and **Lim Kang Yang Leo** (Singapore). The facilitators were Hans L **Rieder** (The Union), Ajay M V **Kumar** (The Union, India), and Jens M **Laurtisen** (EpiData Association).

In **July 2013**, the **forty-seventh** course was conducted in Berne, Switzerland, using EpiData Entry Version 3.1 (270108) and EpiData Analysis Version 2.2.2.182. The course title was “From paper to computer records – ensuring data quality for analysis”. The course was provided for participants enrolled in the Swiss Public Health training curriculum. The participants were Lars P **Andersen**, Anka **Baltensperger**, Nicole **Bartlomé**, Katrin **Crameri**, Corina **Ebnöther**, Nicole **Gysin**, Isabelle **Keel**, Gablu **Kilcher**, Jörg **Lützner**, Miriam **Pfiffner**, Anne **Schmidt**, and Johannes **Wacker**. The facilitators were Hans L **Rieder** (The Union) and Marcel **Zwahlen** (University of Berne, Switzerland).

In **September-October 2013**, the **forty-eighth** course was conducted in Dhaka, Bangladesh, using EpiData Entry Version 3.1 (270108) and EpiData Analysis Version 2.2.2.182. In

addition, an introduction was made to EpiData Manager public release version v1.4.2 and EpiData Entry Client v1.4.2. The course title was “EpiData software for quality-assured data entry and analysis”. The course was provided for personnel working in the Damien Foundation projects in Bangladesh and selected personnel from the national tuberculosis control program. The participants were Muhammad Ameer **Ali**, Panna Lal **Goswani**, Md Anwar **Hossain**, Md Shamin **Hossain**, Aung **Kya Jai Maug**, Priojit Kumar **Nandi**, and Binoy **Tudu**. The facilitator was Hans L **Rieder** (The Union).

In **October 2013**, the **forty-ninth** course was given in Chennai, India, for participants of the National Operational Research Course (2013-14) conducted by the South-East Asia Regional Office of The Union in association with the National Institute for Research in Tuberculosis, India. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.2.182 were utilized. The course was titled “Efficient, Quality-assured Data Capture and Analysis using EpiData”. The participants (all residing in and working for tuberculosis control in India) were Kalpita **Shringarpure**, Deepti **Nirwal**, Sairu **Philip**, Shankar **Dapkekar**, Kiran Kumar **Reddy**, Kumaravel **Ilangovan**, Syed Imran **Farooq**, Hemant D **Shewade**, Umesh Chandra **Tripathi**, Bhavin **Vadera**, Anshul **Avijit**, and Poorana **Ganga Devi**. The facilitators were Ajay MV **Kumar**, Karuna **Sagili**, Deepak **Tamang** (The Union, India), Palanivel **C**, Swaroop **Sahu** (JIPMER, Puducherry), and Mrinalini **Das** (MSF, India).

In **March 2014**, the **fiftieth** course was given in Kathmandu, Nepal for participants of the Asian Operational Research Course conducted by the Centre for Operational Research of South-East Asia office of International Union Against Tuberculosis and Lung Disease (The Union). EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.2.182 were utilized. The participants were Raman **Mahajan** (India), Wajira **Rajapakshe** (Sri Lanka), Sarder Tanzir **Hossain** (Bangladesh), Nay **Thiha** (Myanmar), Nang Thu Thu **Kyaw** (Myanmar), Basant **Joshi** (Nepal), Kimcheng **Choun** (Cambodia), Srinivas **Bairy** (India), Kinley **Zam** (Bhutan), Sai Ko Ko **Zaw** (Myanmar), Aashifa **Yaqoob** (Pakistan) and Vishal **Dogra** (India). The facilitators were Karuna D **Sagili** (The Union, India), Mrinalini **Das** (MSF, India), Prashant **Bhat** (WHO-RNTCP, India), Hemant **Shewade** (Indira Gandhi Medical College and Research Institute, India) and Jay Prasad **Tripathy** (PGIMER, India). Ajay M V **Kumar** (The Union, India) participated as observer.

In **July 2014**, the **fifty-first** course was given in Paris, France, for participants of the fifth Paris Operational Research Course conducted by the Centre for Operational Research of International Union Against Tuberculosis and Lung Disease (The Union). EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.2.182 were utilized. The participants were Riitta **Dlodlo** (Zimbabwe), Thi Mai Phuong **Nguyen** (Vietnam), Janet **Dzangare** (Zimbabwe), Daphne **Lagrou** (Afghanistan/Belgium), Shuisen **Zhou** (China), Mar **Verlade** (Switzerland), Charity **Kanyoro** (Kenya), Mbazi **Senkoro** (Tanzania), Wonda Teshome **Amenu** (Ethiopia), Michel **Sawadogo** (Burundi), Justine **Mirembe** (Uganda) and Josephine **Namboze** (Zimbabwe). The facilitators were Ajay M V **Kumar** (The Union, India), Rafael **van den Bergh** (Luxembourg), Kuda **Takarinda** (Zimbabwe), Philip **Owiti** (Kenya) and C **Palanivel** (India).

In **March 2015**, the **fifty-second** course was given in Kathmandu, Nepal, for participants of the fourth Asia Operational Research Course conducted by the Department of Research, International Union Against Tuberculosis and Lung Disease (The Union), South-East Asia Regional Office, New Delhi, India. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.2.182 were utilized. The participants were Sudeepa **Khanal** (Nepal), Mongal Singh **Gurung** (Bhutan), Hamayoun **Hemat** (Afghanistan), Mahboob **Ul Haq** (Pakistan), Aye **Myat Thi** (Myanmar), Thi Thanh **Huyen Truong** (Vietnam), Sujit Kumar

Sah (Nepal), Zin Min **Thet Lwin** (Myanmar), Kathirvel **Soundappan** (India), Chandor **Wangdi** (Bhutan), and Srinivas **Reddy Edla** (India). The facilitators were C **Palanivel** (India), Vivek **Gupta** (The Union, India), Hemant **Shewade** (The Union, India), Philip **Owiti** (Kenya), Mrinalini **Das** (MSF, India) and Jay Prasad **Tripathy** (India).

In **March 2015**, the **fifty-third** course was given in Cotonou, Bénin, for collaborators from 9 francophone countries in Africa in the observational study of the “Bangladesh 9-month regimen” for multidrug-resistant tuberculosis. The course was given in French but the English course material was used and it focused on the analysis of study data, using as introduction material from Parts A and B. EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version 2.2.2.183 were utilized. The participants were Gisèle **Badoum** (Burkina Faso), Wilfried **Békou** (Bénin) , François **Ciza** (Burundi), Valentin **Fikuma** (République Centrafricaine), Yves **Habimana-Mucyo** (Rwanda), Charlotte **Kangué Tchiche** (Cameroun), Ferdinand **Kassa** (Bénin), Lucien Koffi **Kouakou** (Côte d’Ivoire), Olivia **Mbitikon** (République Centrafricaine), Marie-Léopoldine **Mbulula** (République Démocratique du Congo), Yvon **Ngana** (République Centrafricaine), Jürgen **Noeske** (Cameroun), Ibodé Valéri **Oulai** (Côte d’Ivoire), Alberto **Piubello** (Niger), Tandaogo **Saouadogo** (Burkina Faso), Bassirou **Souleymane** (Niger), Jules **Toloko** (République Démocratique du Congo). The facilitators were Ghislain Kobto **Koura** (The Union), Hans L **Rieder** (The Union), Valérie **Schwoebel** (The Union), and Arnaud **Trébucq** (The Union).

Preparatory steps before you begin

If the course comes on a CD-ROM

If your course is on a CD-ROM, it might be easiest to copy its entire content, i.e., the folder containing the course, to your computer hard disk. One level below the folder, you will see a file:

index.html

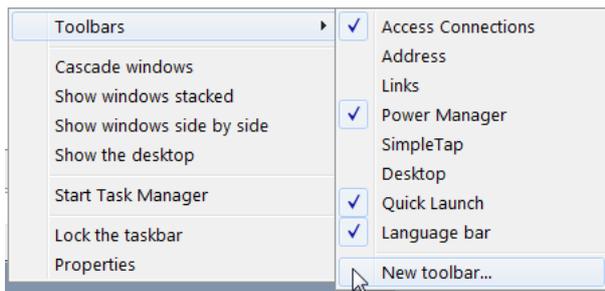
This is the name commonly given by convention to the opening page (the “home page”) of a web site. Indeed, the course material is organized like a web site. If you double-click on this file (INDEX.HTML), the web opens in your default browser. For fastest access to the web you might wish to make a short-cut to this opening page on your desktop and then drag it down to the Quick Launch taskbar (Windows XP® and Windows Vista®). To create a desktop shortcut, right-click on this file, go to Send To and Desktop (create shortcut) and click the latter. You can drag this desktop shortcut down to the Quick Launch taskbar. This will always give you visible access with a single click. Windows 7® has hidden the Quick Launch taskbar and has replaced it with an option to pin programs to the Taskbar. It is possible to restore the old Quick Launch taskbar if you wish (see next paragraph).

Restoring the Quick Launch bar in Windows 7®

Click anywhere on an empty part of the taskbar, right-click and ensure that “Lock the taskbar” is unticked:

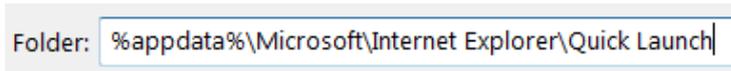


then right-click again and choose “Toolbars” | “New toolbar”:



and type into the location bar:

%appdata%\Microsoft\Internet Explorer\Quick Launch



and press Enter. Ensure that the path is displayed at the top, and click Select Folder. The Quick Launch bar will appear in the Taskbar. Drag the dots to pull it out, then right-click it and untick “Show text” and “Show title”. You can now drag short-cuts from the Desktop into the Quick Launch bar (and then delete them from the Desktop).

Creating working directories

We will require three folders, all in the root of the drive to which you have administrative access, commonly this will be the C: drive:

```
\epidata  
\epidata_course  
\temp
```

The EpiData software, both EpiData Entry and EpiData Analysis, will be installed in the \epidata folder.

In the \epidata_course folder we will keep all our data files that we use and create during this course.

You may already have a \temp folder as it is common and very useful and practice to have a temporary “storage” folder, i.e. a folder where we temporarily store files that we may download from the Internet or indeed in this case from our disk-based course web before we move them to our final destination on our PC.

Installation of EpiData Entry and EpiData Analysis

EpiData Entry is a very small executable file. Although it is Windows®-based, it does not interfere with your Windows® set-up: all files are placed in a single directory (“folder” in Windows® terminology) without leaving any trace of it anywhere else on your system (no *.dll files).

If you have administrator’s rights to install software on your computer

Save the file from the Internet or from the CD-ROM if you are in an in-class course by going to the home page and look for the Software in the navigation panel on the left and click it. In the opening page, look for EpiData Entry and click on the link. In the File download window that opens, click Save (not “Open”). Select your directory of choice [we propose that you use the TEMP folder rather than some Windows® default folder “download” that one may have difficulties to find] and click Save. Repeat the same procedure for EpiData Analysis.

In your file manager (Windows Explorer® or an alternative), find now the EpiData Entry setup file:

```
c:\temp\setup_epidata.exe
```

and double-click the file. You will be suggested the default installation folder:

```
c:\Program files\EpiData
```

While this is perfectly fine, we recommend installing it instead into the root:

```
c:\EpiData
```

typing over the default that is offered and continue to follow the instructions. Of course, you may install EpiData Entry in the default folder c:\Program files\EpiData, but if you work in a place where installation of software is strictly controlled, this might be precisely the folder that is checked for non-approved program files.

The files will be extracted in a breeze. Among the files you see one named EpiData.exe

Right-click on this file, go to Send To and Desktop (create shortcut) and click the latter. Drag the icon from the desktop down to the Quick Launch taskbar. This allows it to be visible from within each program, being just a single click away.

Similarly, install the EpiData Analysis file in the same folder:

```
c:\epidata
```

following the instructions and create a short-cut.

Using EpiData software if you do not have administrator's rights to install software on your computer

Because EpiData software is not interlinked in anyway with the Windows® operating system, an installation is actually not really required. You can download the zip files from the EpiData website, and follow the instructions in the `readme.txt` file or ask your facilitator for assistance. We have also prepared a zip file, available on the CD-ROM, which contains the entire package (EpiData Entry and EpiData Analysis). Just unzip it to the root of the drive to which you have access (or, failing that, any place you have access to).

This is one of the unique things about EpiData installation – it does not require to be installed! All the files required for the functioning of the software are contained in a single folder which can be carried around on a portable memory drive and can be used to begin your work quickly on any computer you have access to. It is an essential principle of EpiData not to interfere with the setup of your computer. (In technical terms: EpiData comes as a few files and does not depend on, install or replace any `.DLL` files in your system directory. Options are saved in an `.INI` file).

Not able to unzip?

If you are not able to unzip the folder, it is likely that you do not have access to a tool which can zip and unzip the files. We have provided freely available, high-quality software (named 7-zip) in the course folder under the navigation panel item Software. Please download and install it. Please ask your facilitator and learn this important skill as you will need it quite often during the course.

Once you have unzipped successfully, you should find a folder `EpiData` in the destination folder. Note though that for apparent reasons, there will be no links to the executable files in the Windows® Start menu: best to make such a link to each of the two executable files on your desktop (then drag to the Quick Launch bar). The two files are:

```
\epidata\EpiData.exe      [to access EpiData Entry]
\epidata\EpiDataStat.exe  [to access EpiData Analysis]
```

Note the corresponding icons for the two softwares.



EpiData Entry

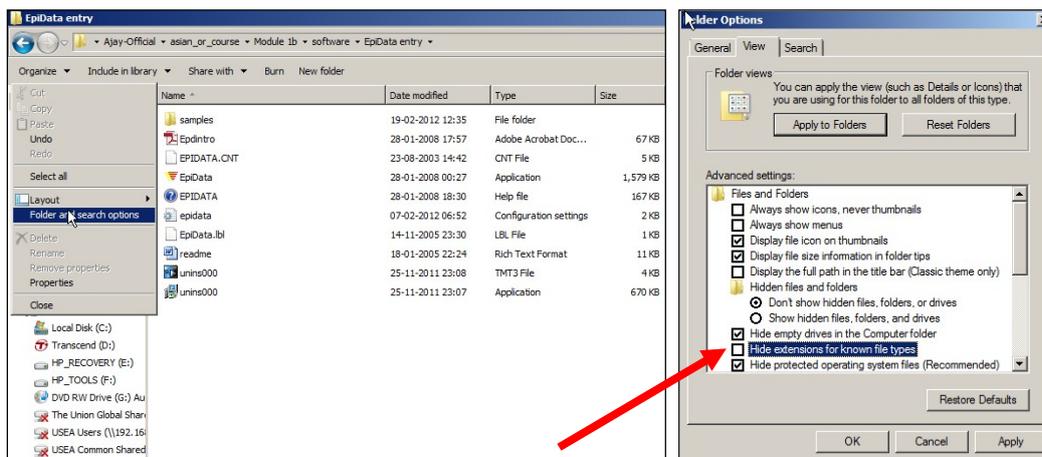


EpiData Analysis

Not able to view the file extensions?

By default and for reasons unknown, the Windows® setting hides the file extensions. However, you will quickly realize that it is essential to change it and make the file extensions visible. The three key files we will be working with in EpiData, namely QES, REC and CHK files (you will know their meaning in the next chapters) can only be distinguished by their extensions. These are called EpiData triplet files and will have the same name for a given set but have different extensions. If the file extensions are not visible, then we will not be able to distinguish between the files. Let us quickly do this.

Look at the screen shots below. Open any folder in your computer. Click on Organize and choose Folder and search options. A new window opens. Click on the View tab and look for the check box with the description “Hide extensions for the known file types”. By default, the check box is ticked. Click on it to un-tick the checkbox. Click on Apply and then OK to come out of the window. You will now notice that the file extensions are visible.



Accessing the EpiData Entry Help file

Windows® has changed the way Help files are being accessed but the necessary plug-in is not part of the operating system in which this change came into effect. As a result, users of Windows Vista® or Windows 7® cannot directly access the EpiData Entry Help file. Windows® expects you to visit its web site:

<http://support.microsoft.com/kb/917607>

and then follow a lengthy process that includes validating that your Windows® version is genuine and then download and install the appropriate plug-in for either 32-bit or 64-bit machines. If you have administrator’s rights, we have a workaround that makes life a bit easier. You need to know two things about your computer:

Vista® or Windows 7®?

32-bit or 64-bit?

If you do not know, then type:

msinfo32

into the Windows® Search bar:



[Enter], and you get both pieces of information:

Item	Value
OS Name	Microsoft Windows 7 Professional
Version	6.1.7601 Service Pack 1 Build 7601
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	WS20110930-1
System Manufacturer	LENOVO
System Model	4174P4G
System Type	x64-based PC

Alternately, you can right-click on My Computer and click on Properties. You will be able to find the system type and operating system. There are four possibilities:

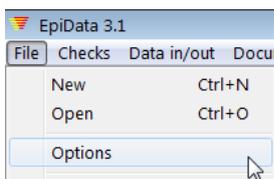
Windows Vista®, 32-bit	requires	Windows6.0-KB917607-x86.msu
Windows Vista®, 64-bit	requires	Windows6.0-KB917607-x64.msu
Windows 7®, 32-bit	requires	Windows6.1-KB917607-x86.msu
Windows 7®, 64-bit	requires	Windows6.1-KB917607-x64.msu

For each of these, you need a different file as shown above. All four are available on the CD-ROM. Download the one applying to you to the `\temp` folder, double-click and follow the instructions. It should work smoothly without prompting to check whether your Windows® is indeed genuine.

Setting Options in EpiData Entry

EpiData software is not laid out to be flashy; its focus is on functionality. Nevertheless, certain options that might appear to be for aesthetics actually serve functionality. We will choose some of the most pertinent options to change the default display. *We must alert you here that you may not understand fully all the options and the reasons for our recommended choices.* However, we suggest that you do as recommended and you will appreciate the detailed rationale as you progress on your learning curve.

Go to “File” | “Options”:



where several tabs allow adaptation of various parts of EpiData Entry:

Note that any change you make here is specific for your PC, and is not retained for another PC unless you also pass on the `EPIDATA.INI` file where these settings are stored. You can also change these settings at any time, including in the middle of data entry. This is particularly handy as the font size you need for entering data might differ from the font size you want to use for a projection in class.

Let us look at the first tab: Editor. Any changes we make here affect the text editor of the software – used in making `QES` and `CHK` files.

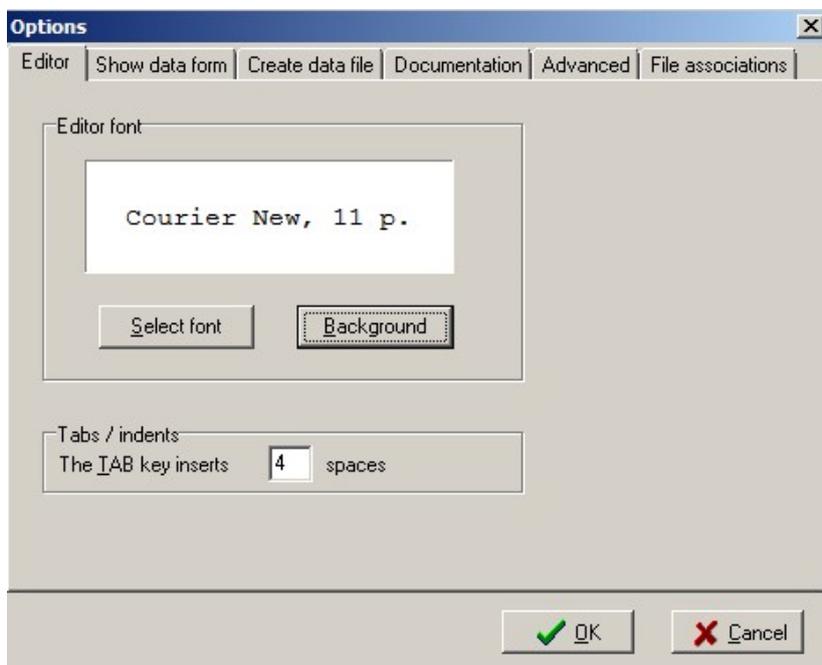
Fonts are important. For our daily life when we read books we prefer a *proportional font*, like one from the *Roman* family. With proportional fonts, each key takes only as much space as the letter requires, as here shown for the difference between a series of ten “l” and ten “m”:

llllllllll
mmmmmmmmmm

While this is ideal for reading a book, it makes life very difficult when writing code in a program. Here we prefer a *non-proportional font*, such as from the *Courier* family, where each letter takes exactly the same space:

llllllllllll
mmmmmmmmmm

In EpiData, we are thus definitely going to make use of a non-proportional font, whatever we are going to do. Chose the font size you think is convenient for you, but stick with Courier.



We recommend that you choose the background color to be white.

Let us now look at the next tab: Show data form. Any changes we make here will affect how the data entry form will look like (REC File). Choose the font and background color as described above.

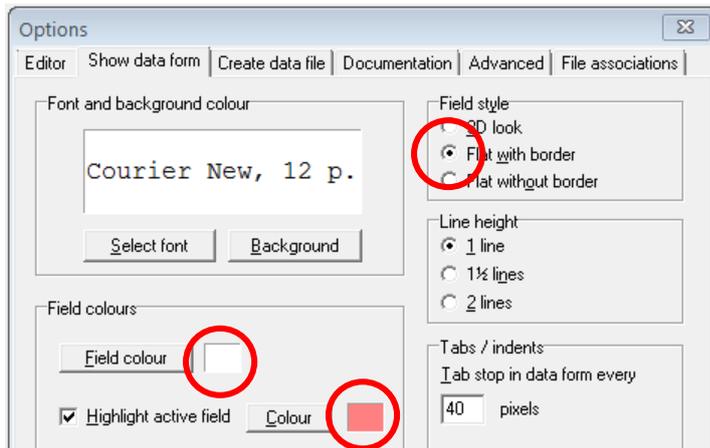
Font for forms: Courier

Background color for forms: White

Field Style (the way data entry fields appear on the form): 'Flat with border'

You may leave the other options under “line height” and “tabs/indents” as it is.

Also important is the choice of the field colors. They shouldn't be straining on the eye. The default of EpiData is actually something that will give you a sore eye after just an hour of work if not earlier. You may make your own choice, but here are three things that we recommend because they have stood the test of data entry:

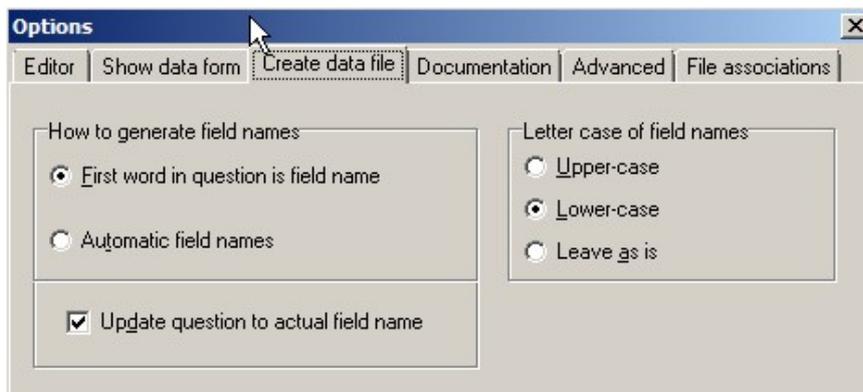


Field color (background color for data entry fields): White

Highlight active field (Shows the currently selected field in a different color): choose purplish color which is in the diagonal of the white for the first:



Let us look at the third and the most important tab: Create data file.

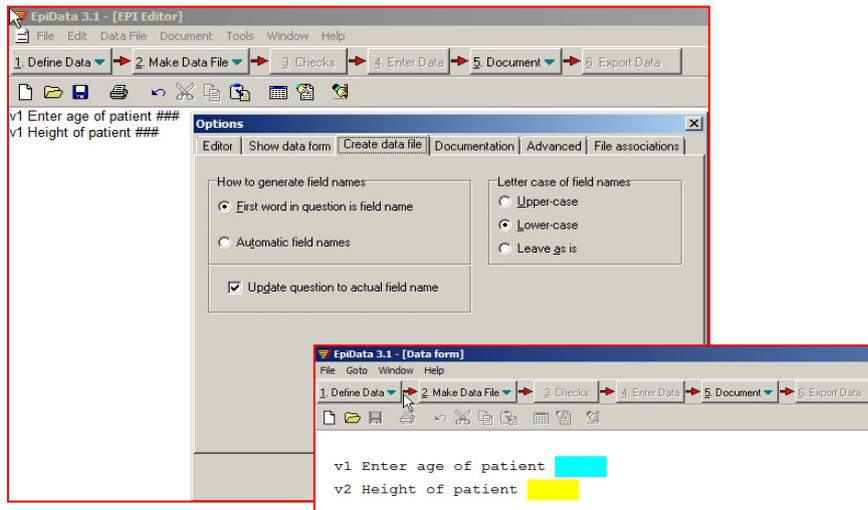


Leave the letter case of field names as “Lower-case”. This will ensure that the variable names will be in lower case irrespective of what is typed when it is exported for data analysis. Though EpiData is not case-sensitive, some other analysis software is (e.g., Stata® and R). Hence, it is a good practice to keep all the field names as lower case.

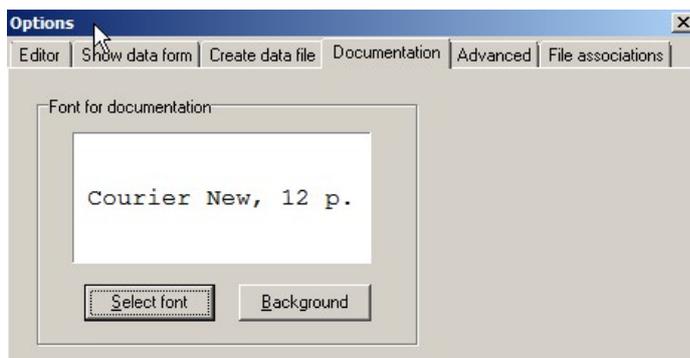
There are two ways of generating a field name or variable name: either let the software generate field names automatically (first eight characters of the sentence) for you or take control and ensure that the first word in every question is the field name. We recommend the latter approach as that helps you to be in command and you can choose your field names and make it as intuitive as possible. *Remember to close all the files before changing options for it to take effect.*

Ticking “Update question to actual field name” will help ensure that every variable has a unique name (one of the important pre-requisites) even though by chance you type the same variable name in the QES file. Look at the screenshot below and carefully note that the

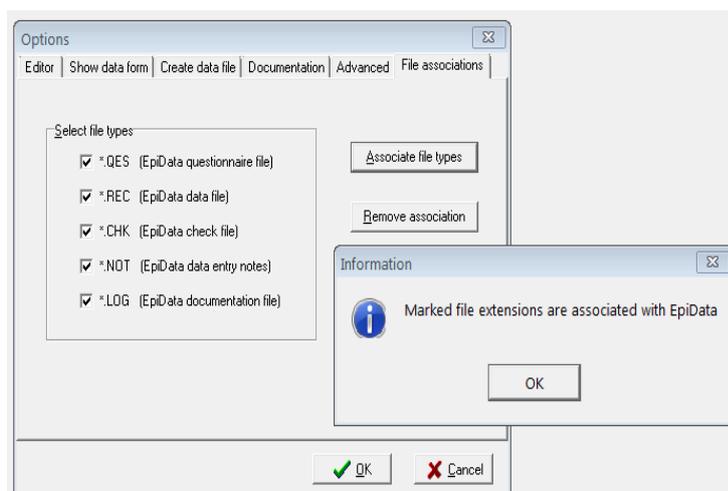
variable name of second variable, though typed as v1 (same name as the first variable) in QES file, it got modified (updated) automatically in the REC file as v2.



The next tab is about setting the font and background of documentation forms (.NOT and .LOG files)



There are some options in the 'Advanced' tab; leave them as they are. Finally, in the 'File associations' tab, tick all the file types and click on 'Associate file types'. This will help open files in EpiData when double clicked in Windows Explorer®.



Which version of the software are you working with?

It is good to know the version of the software that you are working with. This helps in two important uses – First, it will let us know whether we are using the latest version. Second, it will be of immense help when you have some problem and are seeking troubleshooting assistance from your colleague/mentor. Letting your mentor know the version of the software you are using will help your mentor understand the problem and help you better.

How will we know this? It is usually mentioned in the left top corner of the home page of the software. Alternately, you may select Help in the menu bar and click on “About EpiData”. This will describe the software version you are using.

At the end of this chapter, you should have done the following:

1. Installed EpiData entry and EpiData analysis (look for folder EpiData in C: drive)
2. Created desktop shortcuts or quick launch bar icons for both the softwares and checked their functionality.
3. Be able to access and read the HELP file (having installed the appropriate plug-ins)
4. Created working directories in C: folder (named epidata_course and temp).
5. Have set the options in EpiData entry software

If you have done the above steps successfully, we are now ready to move ahead!

Let us begin the exciting journey of EpiData!!

Part A. EpiData Entry

Part A: EpiData Entry

- Exercise 1 A data documentation sheet for a simple questionnaire
- Exercise 2 The QES-REC-CHK triplet
- Exercise 3 Derived fields and Check file commands unrelated to a specific field
- Exercise 4 Data entry and validation
- Exercise 5 Using an external file for Labelblocks
- Exercise 6 Dealing with incomplete dates
- Exercise 7 Keeping track of data entry time
- Exercise 8 Safely backing up and encrypting your data

Acknowledgments:

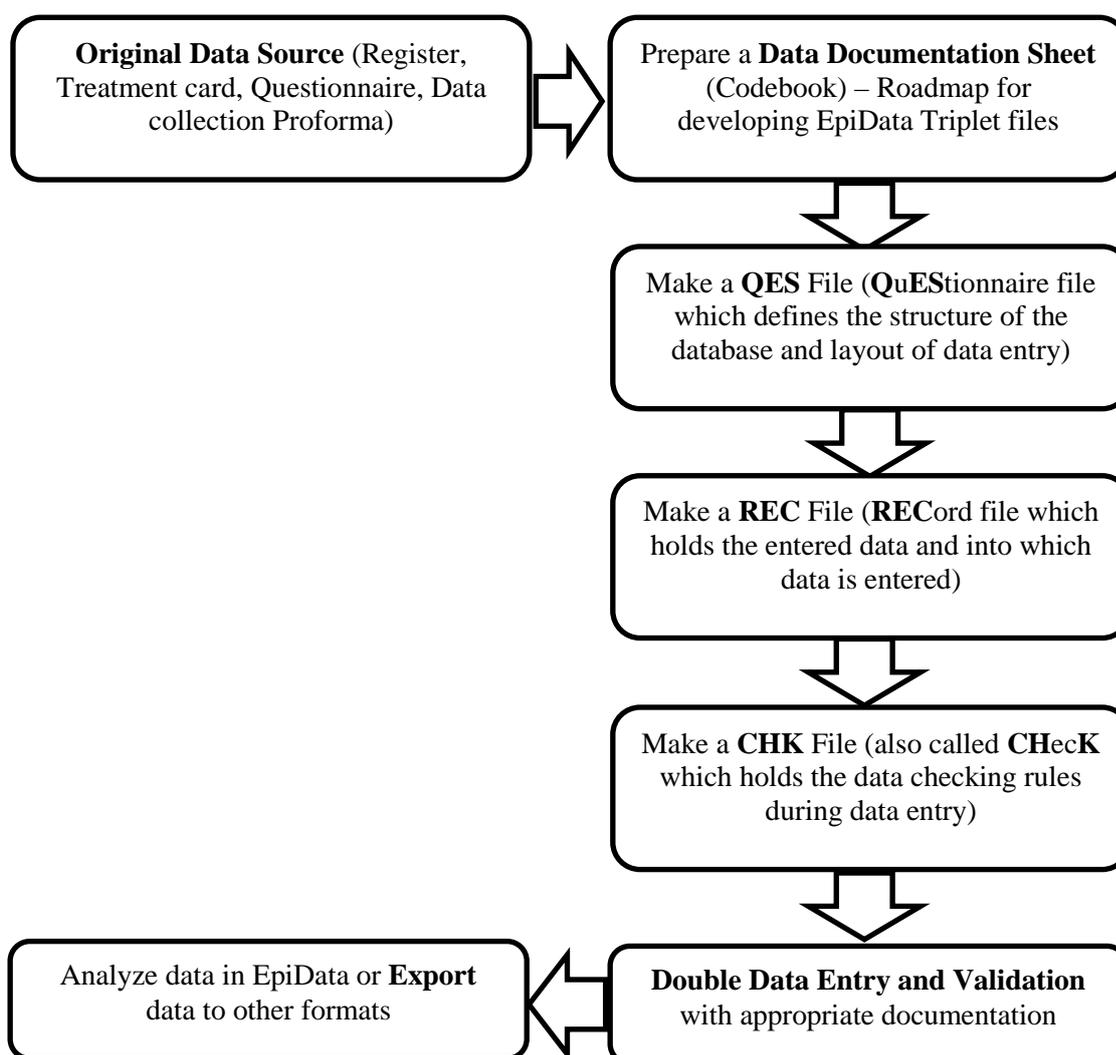
We thank Ajay M V Kumar who has made valuable suggestions to improve the structure and flow of argumentation of Part A.

Exercise 1: A data documentation sheet for a simple questionnaire

At the end of this exercise you should be able to:

- a. Define the different types of fields/variables (text, numeric, date) and know when to use them.
- b. Create a data documentation sheet from a simple questionnaire

Before we proceed to learn the details, let me provide an overview of EpiData entry.



The first step in the process is to prepare a plan for data entry. This plan is called the **data documentation sheet**. This should not be confused with *data collection form* which is the proforma used for collecting the data from study participants or extracted out of the programme records. Data documentation sheet is a codebook containing the details of all the variables (like names, labels, type, length, possible values and value labels) to be entered and the check rules to be applied during the process of data entry. Like Epi Info 6, EpiData Entry uses the same principle of what we call the QES-REC-CHK (pronounced “Ques-Rec-Check”)

files principle. First we create a QES file. This file defines the structure of the database and the layout for data entry including field names, field labels, field type, and field length. From the QES file we then create a REC file (data entry file which will contain all the data), and finally we create a so-called CHK file (which contains and applies the rules of data entry) linked to the data entry file to control data entry. These are referred to as *EpiData triplet files* and are identified by their file extensions (.qes, .rec and .chk). Double data entry and validation is considered a benchmark in assuring data quality and we fully subscribe to this idea. Once QES-REC-CHK files are created, data are entered twice, independently by two persons and compared with each other to identify discrepancies. These are then corrected by referral to the original data source and saved as a final file which is used for analysis. The rationale of double data entry is that the probability of committing the same error in the same field twice, when entered by two independent data entry operators is small and hence data entry errors remaining after validation will be minimal.

Note: Please do not worry if you are not able to understand all the terms at this point in time. Be assured that you will appreciate these as you go along.

But let us proceed step by step and say that we have the following questionnaire:

Laboratory serial number: ____
Date specimen received (dd/mm/yyyy): ____/____/____
Sex: ____
Age in years: ____
Reason for examination: ____
Result of specimen 1: ____
Result of specimen 2: ____
Result of specimen 3: ____

This might present a typical simple questionnaire as used by an interviewer. Often such questionnaires are first completed on paper. This is actually an excerpt from the Tuberculosis Laboratory Register proposed by The Union:

Tuberculosis Programme

Form 2

Tuberculosis laboratory register

Year _____

Lab Serial No.	Date specimen received	Name	Sex M/F	Age	Name of referring facility	Address - patient for diagnosis	Reason for examination*		Results of specimen			Only for SS+ for diagnosis: TB Number or treatment centre**	Remarks
							Diagnosis (tick)	Month of follow up	1	2	3		

We will use this register as the basis for this course. For the time being, you plan to write a short and concise electronic data capture form, retaining only variables that are easy to capture and are likely to be useful for the analysis. *Please note this as a first principle in being efficient – capture only those variables which you will use for analysis!*

Each of the questions can be conceived of as a variable and the answer to the question as the value that the variable takes for a particular individual. **Variables** are also referred to as **'Fields'** in EpiData – both mean the same and will be used interchangeably. We will give

each variable a unique name. A completely entered data form for one study subject is called a **‘Record’**. A set of such records is called a **‘File’** (REC file). The REC file thus contains several records and each record contains information about one individual with respect to several variables. We are going to describe each variable with respect to several attributes in the data documentation sheet. Let us now understand some terminology we are going to use.

- **Field name:** This is the name of the variable and in EpiData, there are certain rules to be followed in arriving at this name. We will come to these rules in a short while.
- **Field Label:** This is the descriptive name for the variable and contains a more detailed description than the variable name can convey.
- **Field Type:** This describes the type of the variable – text, numeric or date being the major types.
- **Field length:** This describes the number of characters that a value can take.
- **Field values:** This describes the possible values that a variable can take.
- **Value labels:** These are descriptive names for the values. For categorical variables which are numerically coded, it is always useful to label them so that it is easier to read and understand what each of the codes mean.

“Labels” are also called **“metadata”** or **“data about data”**. They play a key role in data files. We may have entered a value “9” for a given field, but this number remains meaningless for everyone without clearly specifying for what this value stands. It is important to get acquainted to these terms and understand them clearly since we will be using them frequently. We will be using several examples later in this chapter to clarify these terms.

Field name: Now, let us understand some rules in naming a variable.

First, it has to be **single word** that has **not more than ten characters**. This means that you cannot use a space in the name as a space makes it more than a word. Also, you cannot use any special characters like comma, semicolon, full-stop or underscore.

Note that some other analysis software may accept only a field length of eight characters. If you later plan to export your EpiData files for analysis to such a software package and you had used the full field length of ten, then your field names get truncated.

Second, use a name which is **intuitive** to understand what it means instead of generic field names like v1, v2 and so on.

Third, it may be a good practice to keep the field names in **lower case**. This can be forced by setting up an option. We have already done it and you may verify the set-up in “File” | “Options” | “Create Data File”. Though EpiData is not case-sensitive, some other software is. So, a field name of ‘sex’ (lower case), ‘SEX’ (Upper case) and ‘Sex’ (sentence case) are understood differently. If you later plan to export your EpiData files for analysis to such a software package (two examples are Stata and R, both of which are “case-sensitive”), it may be a problem. Hence, the recommendation to keep it uniformly, “lower case”.

Do not start the variable name with a number. It cannot be ‘1v’, but it can be ‘v1’

The following words ‘date’, ‘month’, and ‘year’ are functions in EpiData and are reserved names. Hence they cannot be used as variable names.

Note: If the Field label begins with a word that is identical to the Field name, you will note later in EpiData Analysis, that this word will be truncated from the Field label. For instance, if your Field name was SEX, and you used 'SEX OF EXAMINEE' as your Field label, this would be truncated to 'OF EXAMINEE'. While this can be fixed easily in EpiData Analysis, it is preferable to prevent it by choosing an alternative Field label during questionnaire design.

Field label: This is the full description of the variable and can be more than a word. Anything that is written between the Field name and the field definition in the QES file is considered as field label.

Field Type: There are different types of entry fields for the variables (we will follow the EpiData Entry notation and call them “Fields”):

- **Text fields:** These fields take letters or numbers or a combination of these as possible values, like PETER, KOCH1882, giraffe, 45677 etc. You can type anything on the keyboard into this field. If you enter a number into such a field, it is accepted but you will not be able to make any calculation with it. These fields are also sometimes designated as character or alphanumeric fields, or most simply “**string**” (denoted by **S**) fields as they take any string of characters.
- **Numeric fields:** These are numbers. The numbers might be integers (denoted by **I**) like 885, 33, 1235 or real numbers like 3.4, 6.88, and 66.5 (also called **floats** and denoted by **F**). You can make calculations with such fields.
- **Date fields:** (denoted by **D**) In different countries, different ways of writing dates are used and this can be confusing for people from another culture. Some write *5 March 2005*, others *March 5 2005*, and again others *2005 March 5*. EpiData Entry lets you choose the type of date you wish to take. In this course we will use European dates, i.e. dates of the format *5 March 2005* or symbolized with DD/MM/YYYY.
- One other type of variables is called “**logic**” or “**Boolean**” variables. This is sometimes used in food-borne outbreak investigations. There, answers to questions on food items eaten is limited to “yes” and “no” and “missing”. In EpiData Entry, this type of field accepts only the values Y, N, 0, 1, and space. There is no need for using this field type, and we actually discourage its use as it might pose problems in analysis. The alternative is a numeric field with a label block.

While you are asked to limit the length of the field name, you have much more flexibility with the length of the value a field can take (up to a field length of 80), but we will try to use it as efficiently as possible, that is we will limit the value length to the minimum needed.

Data Documentation Sheet

It is good practice to write what we call a **data documentation sheet** before you make your actual EpiData Entry QES file. As mentioned earlier, EpiData refers to this as **Codebook**.

In the past, fields like SEX were commonly made text field with values “F” or “M” denoting Females and Males. It is efficient as it uses only a length of 1. Things would get less efficient, if we would have to code treatment outcome with the possible values “cured”, “completed”, “died”, “failed”, “lost from follow-up”, “transferred out”, and “outcome not recorded”.

Numeric coding is much simpler as there are up to ten possible values with a field length of just 1:

- 1 Cured
- 2 Treatment completed
- 3 Died of any cause
- 4 Failed bacteriologically
- 5 Lost from follow-up
- 6 Transferred out
- 9 Outcome not recorded

You will also realize later in the analysis that this will be very convenient to apply the selection criteria when you want to select a subset of data and undertake analysis only on the subset. Of course, a prerequisite is that the link between the numeric value and the text label is unambiguously clear. The role of labels is of enormous importance, they are also called meta-data or “data about data”. We are going to make full use of numeric coding of field values and using explicit text as value labels.

Let us now go through a few examples of the variables (from the tuberculosis laboratory register example) and describe the various attributes of the variables in the data documentation sheet. As you go through, you will note that preparation of data documentation sheet requires thinking and knowledge of study data.

This is how we would write such a data documentation sheet:

Field name	Field label	Field type	Field length	Field values	Value labels	Comment
serno	Laboratory serial number *	I	4	1-9000, 9001, 9002,		Serial number starting with 1 each year Enter 9001, 9002,... if serial number is <i>not unique</i> or <i>missing</i> , and write a data entry note (use F5 to open a note file)
regdate	Registration date	D	10	01/01/2000-31/12/2005, 01/01/1800		Range of legal registration dates Enter 01/011800 if no date recorded
sex	Examinee's sex	I	1	1 2 9	Female sex Male sex Sex not recorded	

* **Note:** Commonly, it will be preferable to make the identifier a text field. If it is a number, as in this case here with the laboratory serial number, precautions must be taken to distinguish e.g. “0001” from “1”, requiring that the numeric value is entered into one field, and another field, the actual identifier field, is automatically correctly calculated to add leading zeros where appropriate. This will actually be done in a later exercise.

Task:

- o Complete the data documentation sheet for all fields in the questionnaire. Note that you should always define a value if no answer was provided to a question.*
- o Think of the most efficient ways to code reason for examination and results of microscopic examination*

Solution to Exercise 1: A data documentation sheet for a simple questionnaire

Key Point(s):

- It is good practice to write a data documentation sheet before you make your actual EpiData Entry QES file.
- You should always define a value if no answer was provided to a question.
- “Date” is a reserved name in EpiData and cannot be used as a field name.

Task:

- o Complete the data documentation sheet for all fields in the questionnaire. Note that you should always define a value if no answer was provided to a question.*
- o Think of the most efficient ways to code reason for examination and results of microscopic examination.*

Solution:

There are many different solutions, but for the sake of uniformity, we will be using the following (but later revise some components of it) as shown on the next page.

Field name	Field label	Field type	Field length	Field values	Value labels	Comment
serno	Laboratory serial number	I	4	1-9000 9001, 9002,		Serial number starting with 1 each year Enter 9001, 9002,... if serial number is <i>not unique</i> or <i>missing</i> , and write a data entry note (use F5 to open a note file)
regdate	Registration date	D	10	01/01/2000-31/12/2005, 01/01/1800		Range of legal registration dates Enter 01/01/1800 if no date recorded
sex	Examinee's sex	I	1	1 2 9	Female sex Male sex Sex not recorded	
age	Examinee's age in years	I	3	0-125, 999		Range of legal years Age not recorded
reason	Examination reason	I	1	0 1 2 3 4 5 6 7 8 9	Diagnosis Follow-up at 1 month Follow-up at 2 months Follow-up at 3 months Follow-up at 4 months Follow-up at 5 months Follow-up at 6 months Follow-up at 7 months or later Follow-up, month not stated Reason not recorded	
res1	Result of specimen 1	I	1	0 1 2 3 4 5 6 9	Negative 1+ positive 2+ positive 3+ positive Positive, not quantified Scanty, not quantified Scanty, quantified Result not recorded	<i>[If the entered value is other than 6, then write 0 into the next field and bypass it]</i> <i>[If the entered value is 6, then ensure that 0 cannot be entered into the next field]</i>
res1sc	Result of specimen 1 scanty	I	1	0 1 2 3 4 5 6 7 8	Not applicable 1 AFB per 100 OIF 2 AFB per 100 OIF 3 AFB per 100 OIF 4 AFB per 100 OIF 5 AFB per 100 OIF 6 AFB per 100 OIF 7 AFB per 100 OIF 8 AFB per 100 OIF	

				9	9 AFB per 100 OIF	
res2	Result of specimen 2	l	1	0 1 2 3 4 5 6 9	Negative 1+ positive 2+ positive 3+ positive Positive, not quantified Scanty, not quantified Scanty, quantified Result not recorded	<i>[If the entered value is other than 6, then write 0 into the next field and bypass it] [If the entered value is 6, then ensure that 0 cannot be entered into the next field]</i>
res2sc	Result of specimen 2 scanty	l	1	0 1 2 3 4 5 6 7 8 9	Not applicable 1 AFB per 100 OIF 2 AFB per 100 OIF 3 AFB per 100 OIF 4 AFB per 100 OIF 5 AFB per 100 OIF 6 AFB per 100 OIF 7 AFB per 100 OIF 8 AFB per 100 OIF 9 AFB per 100 OIF	
res3	Result of specimen 3	l	1	0 1 2 3 4 5 6 9	Negative 1+ positive 2+ positive 3+ positive Positive, not quantified Scanty, not quantified Scanty, quantified Result not recorded	<i>[If the entered value is other than 6, then write 0 into the next field and save record to disk] [If the entered value is 6, then ensure that 0 cannot be entered into the next field]</i>
res3sc	Result of specimen 3 scanty	l	1	0 1 2 3 4 5 6 7 8 9	Not applicable 1 AFB per 100 OIF 2 AFB per 100 OIF 3 AFB per 100 OIF 4 AFB per 100 OIF 5 AFB per 100 OIF 6 AFB per 100 OIF 7 AFB per 100 OIF 8 AFB per 100 OIF 9 AFB per 100 OIF	

Note the following here. For an unknown laboratory date (REGDATE), we must enter a legally existing (valid) date. EpiData will not accept a date 99/99/9999 nor for that matter 29/02/2001. It is a personal preference of us to usually use 9 or 99 . 9 or the like to define unknown values, be this is only for text or numeric variables. We also introduced a “legal range” for some variables like REGDATE and AGE. We did this a bit arbitrarily, but still tried to keep it within what might be expected.

We made two fields for each result. There are 17 possibilities for a result, and therefore a length of 2 is the minimum required, but even with that the values might not be intuitive, but they should be. An alternative version uses a float of length three to get for instance:

```
0.0    Negative
1.0    1+ Positive
2.0    2+ positive
...
0.1    Scanty, 1 AFB per 100 OIF
0.2    Scanty, 2 AFB per 100 AFB
```

Scanty results are relatively rare among positives (perhaps some 10% among diagnostic and some 20% among follow-up examinations in quality-assured high-burden country laboratories), and positives themselves are relatively rare among all (perhaps 10% to 20% among patients coming for diagnostic evaluation). Thus scanty positive results might be only 1% of all results. We therefore chose to use integer variables and bypass the field scanty, unless the first field defines the result as quantified scanty.

Exercise 2: The QES-REC-CHK triplet

At the end of this exercise you should be able to:

- Create and edit a questionnaire file (*.qes).
- Make a record file (*.rec).
- Make and edit a check file (*.chk).
- Understand the QES-REC-CHK triplet and how the three are related to each other.

You are now ready to start with the design of the questionnaire in EpiData Entry, based on the data documentation sheet.

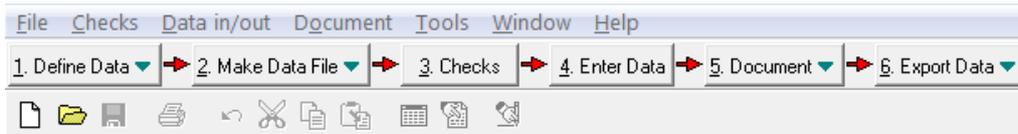
Before starting, let's clarify the terminology and where what is going to be placed. There are four elements that go with a field:

Field name	Field label	Field value	Comment / Value label
age	Person's age in years	<input type="text" value="23"/>	Enter 999 if not recorded
sex	Person's sex	<input type="text" value="9"/>	Not recorded

We call the variable the **Field name** (see Exercise 1). EpiData Entry interprets the first word in a line to be the Field name (see options). To have more explicit information than what can be conveyed by the length 10, single-word Field name, we supplement the Field name with a more explanatory **Field label**. Anything between the first word (Field name) and the Field definition is interpreted by EpiData to be the Field label. The third element on the line is the Field definition which receives the actual **Field value** during data entry. The field value is the actual datum (singular of “data”) for that variable for that observation. For *continuous variables*, we might add an explanatory **Comment** to the right of the field definition, here for instance what to do if the primary source has no information on this variable for this observation. For *categorical variables*, the comment is not necessary: a label block will comprehensively provide information on what is entered. As we use numeric coding, it would, however, be a nice touch if after picking a value from a pick list (label block), the **Value label** would appear to the right of the field, as a kind of confirmation that what was picked was what was actually intended.

In the data documentation sheet we made two columns to distinguish “Comment” (for continuous variables) and “Value label” (for label blocks with categorical variables).

Open EpiData Entry by double-clicking the icon on your desktop or single-clicking the icon in your quick-launch task bar. You see the EpiData Entry has three rows on top of the screen - the top row (**Menu Bar**) with File Checks Data in/out etc, the second with a numbered sequence (**Work Process toolbar**), and the third (**Editor toolbar**) with mostly grayed out icons:



For the time being we concentrate only on the middle row that shows the following sequence (this is called the “Work Process toolbar”):



Each of these has a menu which you see when you click on the box. You can see immediately where you have to start.

1. Define Data | **Step 1: Creating the *.QES file**

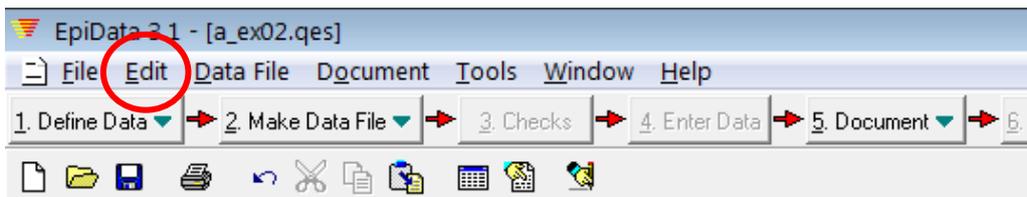
If you click on “1. Define data” (or using the shortcut **ALT+1**) the menu with two options pops up: “New .QES file” and “Open .QES file”. EpiData questionnaire files have the extension “*.QES”. As you must now create a new questionnaire file and not open an existing QES file, click “New .QES file” and the empty screen ready for writing opens.

Start to type like in any word processor the following:

```
This is the questionnaire for the laboratory register
serno Laboratory serial number
```

Let’s save this right away as **A_EX02.QES** (shortcut: **CTRL+S**).

We have to associate two things with this field, the type of field and the field length. In the top menu line you see “Edit”.

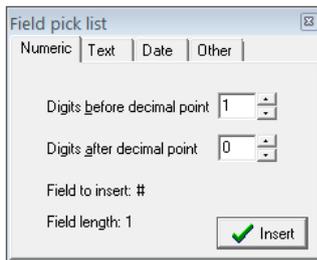


```
This is the questionnaire for the laboratory register
serno Laboratory serial number |
```

Note: While EpiData Entry is not case-sensitive (that is, you may use upper-case or lower-case), some statistical packages are case-sensitive (“sex” is not identical to “Sex” or “SEX”). You are on the safe side if you make it a habit of using always lower-case for field names. See also “File” | “Options” | “Create Data File” to force lower-case for variable names.

Click on it to get the drop down menu. However, we encourage you to learn the shortcuts, they are often more efficient than the mouse. If you click the **Alt** key, you see that each of these menus has one letter underlined. In this case it is the E in Edit. Thus **Alt+E** is a fast way to see the drop down menu. In it, you see “Field pick list **CTRL+Q**”. Pick it by typing “f”

twice in sequence and the Field pick list box opens. You see four tabs, Numeric, Text, Date, and Other:



Numeric fields

In numeric fields you define the number of digits before the decimal point. If there is none (zero) after the decimal point, the numeric field is an integer. If there is 1 or more digit after the decimal point, the numeric field is a float (real number). The symbol is the # (hash) sign:

Integer of length 3

#.# Float of length 4 (the decimal point contributes to length)

For the field SERNO, we defined it to be an integer of length 4. Make sure that you have a space between the field name and the field type / field length:

```
serno Laboratory serial number ####
```

We add as per data documentation sheet the comment after the field definition what to do if one encounters a duplicate serial number.

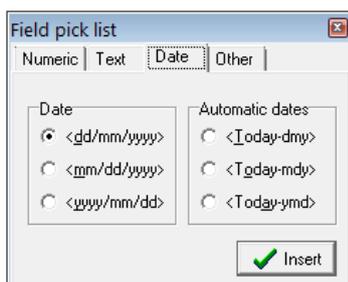
Text fields

Text: using this type you may enter upper or lower case letters or a mixture of cases, and the value will be exactly as you entered it. Commonly, these fields are also called “**String**” fields, as we may enter any string of characters that can be found on keyboard into such a field.

Upper-case text: if you enter a lower-case letter from your keyboard, it will automatically be converted to upper case. This is often very useful as the field value “m” is not the same as “M” and these would be considered as separate categories when you run a frequency table in analysis.

Encryption field: this is a powerful tool if you enter information like personal identifiers that should be openable only by persons with access to a password. More on data encryption will be discussed later in this course.

Date fields



On the left hand side, we see that we have three options to visualize dates:

dd/mm/yyyy	“European” date
mm/dd/yyyy	“American” date
yyyy/mm/dd	“Chinese” date

Note: It is of critical importance that you choose the style that is used in your setting else it is likely that many data entry errors will be made. To do it, click on start-control panel-regional and language options-choose English (UK). Also, note that calculations by EpiData are made with European dates, what is chosen here is how dates are displayed (and entered), not how calculations are actually made internally.

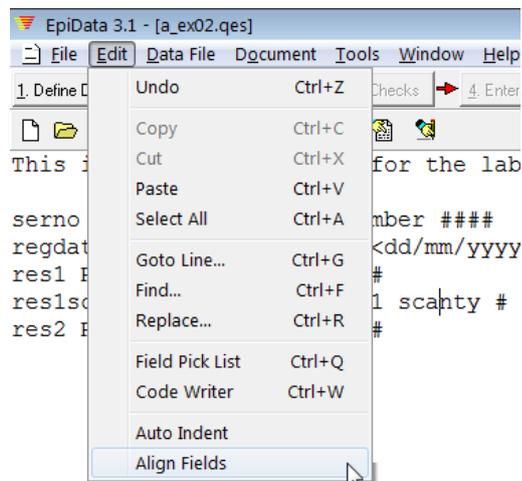
We are now only partially completing the questionnaire file, for the additional fields REGDATE, RES1, RES1SC, and RES2. For the remaining fields you will be doing as part of the task for this exercise. Thus, adding these four additional fields, you get:

This is the questionnaire for the laboratory register

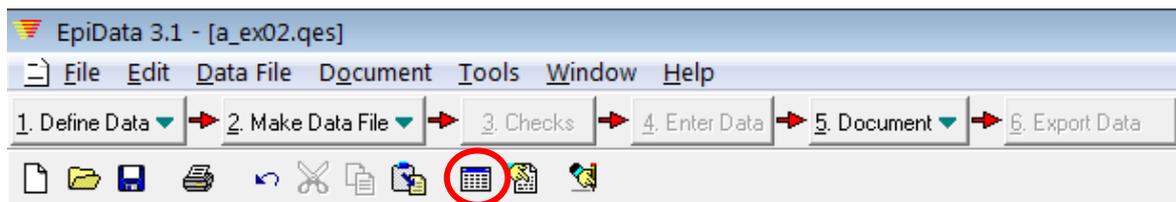
```
serno Laboratory serial number ##### Enter 9001, 9002, ... if serial
regdate Registration date <dd/mm/yyyy> Enter 01/01/1800 if missing
res1 Result of specimen 1 #
res1sc Result of specimen 1 scanty #
res2 Result of specimen 2 #
```

Aligning the fields

As you can observe, the fields are not properly lined up. Move your cursor anywhere to the field with the longest text before the field definition, here this is the line for the field RES1SC, then pick again Edit from the top line to see “Align fields” as the last on the drop-down menu and click it:



then the fields get aligned (sometimes you need to do it twice). To see how it will look for the data entry person, click the third icon from the right in the bottom third line:



and you get:

This is the questionnaire for the laboratory register

serno	Laboratory serial number	<input type="text" value="9001"/>	Enter 9001,
regdate	Registration date	<input type="text"/>	Enter
res1	Result of specimen 1	<input type="text"/>	
res1sc	Result of specimen 1 scanty	<input type="text"/>	
res2	Result of specimen 2	<input type="text"/>	

F10 or **CTRL+F4** gets you out of this preview. For the time being, we leave this incomplete `A_EX02.QES` file and proceed with the next step.

2. Make Data File Step 2: Making the *.REC file

We have now the QES file, and this provides the information on field definitions for the data file. **ALT+2** opens the drop down menu to pick the Make data file sub-menu which opens the “Create data file from *.QES file” dialogue box. The name of the *.QES file is already there and the same file name (with the *.REC extension) is proposed. That is very sensible, it should be so, and it must be so, and we accept this. **Note this very important principle – the names of the QES and REC files should have the same name to be linked with each other.** We are now prompted to enter a description. This is not necessary but it is helpful documentation. Thus we type in for instance “Exercise 2” and we are informed that the `A_EX02.REC` has been created.

Actually, this is enough to start data entry! You can open the REC file and start entering the data. However, you will notice that fields now can receive any values. The field definition itself poses some basic checks; for example, you will not be allowed to enter a text in a numeric field or you will not be able to enter an illegal date in a date field, but it can only go so far! *Rather than checking the data after all data have been entered, it may be useful to check the validity of the data during the data entry process.* Using a Check file (same name as the REC file but with the extension .CHK instead of .REC) makes this possible. It is in the CHK file where you define all the rules of data entry control. CHK files can do a range of things in ensuring data validity and include range checking, specification of legal values, making a field required, making conditional jumps between fields or any other conditional operations, using value labels, giving messages to the data entry operator and making complex calculations from the values entered in other fields.

Let us create a CHK file right away!

There are two ways of creating a CHK file: By using menu-based option or by using the editor and writing all the CHK commands manually. While both is possible, preferentially the menu should be used for all field-level checks, because EpiData checks here your grammar. Let us thus begin with the menu and postpone the learning of how to edit the CHK file in a text editor when we create Checks that apply to the record or the file level (rather than a specific field), where using the menu is not possible.

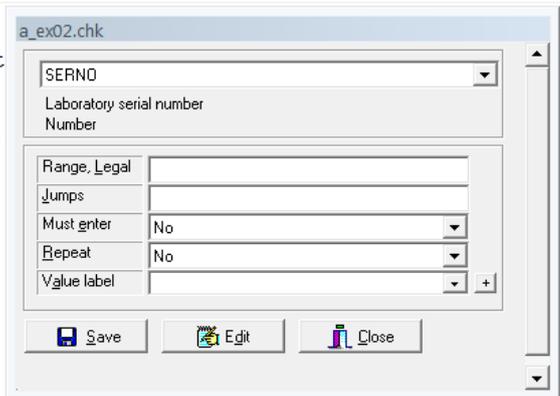
Step 3: Making the *.CHK file

With **ALT+3** we open the dialog box “Select data file for checks”. The A_EX02.REC is suggested and that is the file we need.

The entry screen appears with one variable and a box showing that we have indeed now the third file A_EX02.CHK:

This is the questionnaire for the laboratory regist

serno	Laboratory serial number	<input type="text"/>	Enter
regdate	Registration date	<input type="text"/>	
res1	Result of specimen 1	<input type="text"/>	
res1sc	Result of specimen 1 scanty	<input type="text"/>	
res2	Result of specimen 2	<input type="text"/>	



The cursor is in the first field SERNO and we can see that this is a “Number” (numeric) field.

All Checks added with:

refer to the specific field which has the focus during data entry. **To add checks with reference to a specific field, you have to select the field by moving the cursor into that field.** Right now, the cursor is in the field SERNO (which is highlighted) and any checks you add will be for that field.

In the pop-up box you see different types of Checks we can apply to the field on which the focus is:

Range, Legal Here you can enter all legal values, i.e., it defines what values the data entry person is allowed to enter. In the case of the unique identifier, there will, by definition, be as many as there are individuals on whom information has to be entered. It does therefore not make any sense to define legal values for the variable SERNO. **Legal values are used for continuous variables. Do not use Legal values for categorical variables.** We have two fields for continuous variables, the registration date REGDATE and the patient’s AGE:

01/01/2000-31/12/2005, 01/01/1800
0-125,999

You can only have a single range, but you may add multiple single legal values, separated by commas.

Jumps You could determine here if a subsequent field should be skipped if the current value takes a certain value. For instance, you may have the value “2” for male and the value “1” for female in the field SEX. If the person is female, one might ask how many pregnancies (variable, e.g., PREGNO) she has had, but if the person is male, one would obviously not ask this question. Thus, you would

jump (bypass) the question about pregnancies in case the study subject is male and go in that case to the field after pregnancies which might be AGE. To tell that a jump is needed in case the value of the field SEX is “2”, you would simply type:

```
2>age
```

In the case of the field SERNO, no jump is needed, and we will leave this open. We actually prefer an alternative to Jumps as shown specifically below.

Must enter Here you define whether information must be entered or not. If you define a field as “MUSTENTER”, then once you enter into the field, you will not be able to get out of the field without entering a value. In the case of the field SERNO, we will require that it is entered. Choose thus “Yes” from the drop-down menu at the right. You will therefore not have any missing values for this variable. *Note: in this course, we will always use MUSTENTER fields* (except for automatically calculated variables).

Repeat Here you can specify whether a value for the field should be repeated in all subsequent records, unless you choose to overwrite the value. This comes in handy when you make a field for capturing district name, and the district name in the field is the same for a set of consecutive records. For the field SERNO this is obviously not the case.

Value label If we code, e.g., the value “female” for the field SEX with “1” and the value “male” with “2”, then it could prove useful to label it so that an explanation appears that “1” stands for “female” and “2” for “male” to reduce data entry errors. It will make the life of the data entry person so much easier.

You are reminded to save by clicking on ‘Save and close’ after working on checks.

Note: As you choose some check options in the pop-up menu, commands are written in the background. For example, if you click on Edit, you will notice the following:

```
serno
  MUSTENTER
END
```

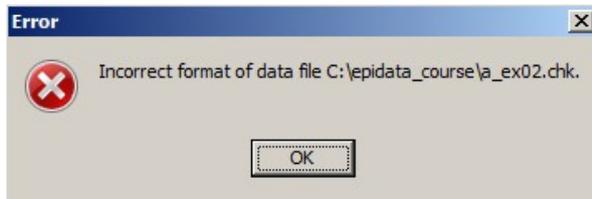
As you notice, the command MUSTENTER is written as we chose the option in the menu. Alternately, we can write MUSTENTER here directly and it is the same as choosing the menu option. Add and delete here to observe what happens on the menu. We recommend that you notice as you choose options from the pop-up menu.

Similarly for regdate, the following will appear as we define the range and legal values

```
regdate
  RANGE 01/01/2000 31/12/2005
  LEGAL
    01/01/1800
  END
  MUSTENTER
END
```

Note: When you are adding or revising checks in this way, that is via ‘3. Checks’, you have to click on the REC file for which the checks have to be defined. Sometimes, going by intuition,

if you try to open a CHK file through this path by selecting the CHK file, you will encounter an error:



At first it might appear counter-intuitive. If we think about it a bit, it becomes clear that we create here controls of what can be entered into a data (i.e. REC) file: we “make Checks for a REC file”. We will introduce to you another way of opening the CHK file where you will follow the usual method – click on the CHK file to open it.

Specific points, exemplified with the five fields

MUSTENTER

We will make every field a MUSTENTER field, unless it is an automatically calculated field. The reason for this choice is that leaving a field empty could mean two different things:

- 1) The field in the original data source was empty;
- 2) the data entry person forgot to enter the information.

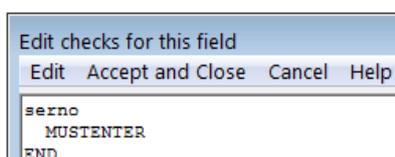
These two are obviously fundamentally different. To reduce the risk of the second, we make each field a MUSTENTER field. This can be done by clicking and selecting or with the short-cut key combination **CTRL+E**. For this exercise, there are no calculated fields, all will thus have to become MUSTENTER.

KEY UNIQUE

The most important variable in any dataset is a unique identifier for every record. The identifier must be unique because if it is not, it will not ever be possible to identify the record with certainty. The recommendation for the tuberculosis microscopy laboratory register is that the laboratory serial number is to start with 1 each calendar year and be strictly sequential throughout that year. It is thus a unique identifier for each record in a given year in a given laboratory. Often, identifiers are not that neatly sequential and it will then not be possible to easily ascertain whether the identifier had already been used earlier in the data set. We therefore need EpiData to check after entering the identifier (here the serial number) in every record whether it had been used before or not and if so to report where and to prevent the data entry person to continue to work before the problem is addressed. As you might have noticed, there is no fixed menu option to make a variable “unique”; thus, we have to write a command. At the bottom of the Check file box, you can see “Edit”:



If you select “Edit” we get:



with the field `serno`, followed by the command `MUSTENTER` in the second line and finishing with the command `END`. We add here:

```
serno
  KEY UNIQUE 1
  MUSTENTER
END
```

If we don't number the `KEY`, EpiData will do that. Using keys will make EpiData to automatically create an index file with the same name as the `*.QES`, `*.REC`, and `*.CHK` file, but with the extension `*.EIX` (EpiData Index file). Index files are single-purpose files. In this case, the only purpose is to keep the information on all `SERNOs` ever used and to determine after a new one has been entered whether it had not been used before. If it has, it must raise an alert with information in which record it had been used. This single-purpose action makes index files very fast in search function and the user doesn't even notice that they do their job except if there is a problem.

RANGE, LEGAL

The legal registration dates are constrained to be allowed only from 1 January 2000 through 31 December 2005 plus a value of 1 January 1800 if the registration date is not recorded. The range is indicated by a hyphen separating the minimum and the maximum, and legal values follow, separated by commas, thus:

```
01/01/2000-31/12/2005,01/01/1800
```

Is entered into the appropriate space:

Range, Legal	01/01/2000-31/12/2005,01/01/1800
--------------	----------------------------------

Note that you cannot define more than one range!

LABELBLOCK

The field `RES1` is a variable with results expressed categorically and has thus been defined as an integer field of length 1. If we look at the Check box:

```
res1      Result of specimen 1  Value label 
```

We see on the last line "Value label" and a plus sign at the end of the line. If you click on the plus sign you get:

```
Edit value labels
Edit Accept and Close Cancel Help
LABEL Label_res1
END
```

EpiData has prepared the beginning and end of the label block for this field. The command `LABEL` is followed by a proposed Label name `Label_res1` for the label and the command finishes with an `END`, while the cursor is blinking in between, ready for you to write. The name for the label is a suggestion, and commonly it makes sense to just use it, but in this case, we will choose to modify it and will call it `label_result`:

```

LABEL label_result
END

```

This is not necessary. It is on one hand just to show that you may modify the proposed name, but also because there are in the end 3 results for which we can use the same label. While it is not wrong, it might look a bit strange to use `label_res1` for RES2 and RES3.

We now faithfully type into the space between LABEL and END exactly what we have defined in the data documentation sheet:

```

Edit value labels
Edit Accept and Close Cancel Help
LABEL label_result
0 Negative
1 "1+ positive"
2 "2+ positive"
3 "3+ positive"
4 "Positive, not quantified"
5 "Scanty, not quantified"
6 "Scanty, quantified"
9 "No result recorded"
END

```

Note that `Negative` is not in quotation marks, but all other Value labels are. The rule is that a label consisting of more than one word (defined as a space character separating two strings) must be put into quotation marks. Single-word labels can but don't need to be placed in quotation marks.

We accept and close (**ALT+A**) and can see now the label that is being used for the field:

specimen 1  Value label label_result

SHOW and TYPE COMMENT

Below the line Value label, we see the tab Edit:

Upon clicking, we get:

```

Edit checks for this field
Edit Accept and Close Cancel Help
res1
COMMENT LEGAL USE label_result
MUSTENTER
END

```

This is the actual paragraph written into the Check file. Let's take it apart.

The paragraph starts with the field name, `res1`, and finishes with `END`, and then there are two lines of commands in between. This is the principle for Checks for any field in EpiData: if there are any Check commands, then there is first the field name and it concludes with an `END`. If there are no Check commands, then nothing is written.

The **indent** has no functional value but it greatly facilitates a quick ascertainment where something (the field) begins and where it ends.

In between we have two lines of commands. Again, **commands are capitalized**, but the name of the label or the name of the field is not: lower case or capitals have no functional meaning: **EpiData is not case-sensitive**. It is for visual guidance only that EpiData capitalizes all commands but not names of fields and labels. It helps to quickly make distinctions. As a user you will not have to follow this convention, but it is actually tremendously useful and there is nothing wrong with adopting these conventions.

Now what do these commands mean? The first line has the command `COMMENT LEGAL USE` followed by the name of the label we just created before, while the label block is actually nowhere to be seen. We mentioned above that `RANGE` and `LEGAL` should not be used for categorical variables, and perhaps now it is becoming clearer why not: instead of legal values indicated in the line above:

Range, Legal	
Jumps	
Must enter	Yes
Repeat	No
Value label	label_result

we use a label block to provide what is legal to use and this is expressed by this command `COMMENT LEGAL USE label_result`: it is legal to use the label block `label_result`. We cannot see the actual label block here: EpiData keeps all label blocks separately in one “super-block” at which we will look in a later exercise. Suffice it here to know that this command refers to it.

You recognize `MUSTENTER`: by choosing that all fields including the field `res1` are Mustenter fields, EpiData recorded this as a separate command on a second line.

The command `COMMENT LEGAL ...` defines that no value other than what was defined in the label block is legal. This is not quite sufficient for our convenient use. What we want is that when entering the field `RES1`, a Label block pops up from which we can pick the pair of field value and field label as here:

The default of EpiData is not to show the label block. To invoke the labelblock during data entry and make it show up requires adding the command `SHOW` after the name of the label block:

```
COMMENT LEGAL USE label_result SHOW
```

We also wanted that the value label is actually (temporarily) written to the right of the field after entry:

Result of specimen 1 Negative

This requires the additional command `TYPE COMMENT` written into a new line:

```
res1
```

```

COMMENT LEGAL USE label_result SHOW
TYPE COMMENT
MUSTENTER
END

```

AFTER ENTRY

There is one more issue to be addressed. We have a second variable for each result, and that is a field to be completed only if the result is ‘scanty, quantified’. The value we give for a quantified scanty result in the field RES1 is “6”. In other words, only if the value in RES1 is 6 should the cursor go into the field RES1SC. In any other case, it should enter the value 0 into the field RES1SC, skip the field and place the cursor into the field RES2. This is where the command AFTER ENTRY comes into play. This is a block command and *specifies a block of commands that are executed after a value has been entered in this field. This is a very useful command for making calculations in the CHK file.* AFTER ENTRY must be terminated with an END by syntax, and we thus have to make use of space between AFTER ENTRY and END to enter the commands to be executed:

```

res1
  COMMENT LEGAL USE label_result SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    Something goes in here
  END
END

```

What needs to be entered in between the AFTER ENTRY ... END statements is conditions of the type “If this is the case, then do that”. In EpiData, this is done with:

```

IF ... THEN
  Something to happen
ENDIF

```

We note here three things: first, the IF and THEN are on the same line; second, the thing or things that should happen are on another line, and third, an IF command finishes with an ENDIF. We thus get in extension:

```

res1
  COMMENT LEGAL USE label_result SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    IF ... THEN
      xxxx
    ENDIF
  END
END

```

The “xxxx” doesn’t need to be a single command, it can be multiple ones, each written into a separate line. Specifically, what we want is that if the value that was entered for the field RES1 is not equal to 6, then the value for the field RES1SC should become 0 and the cursor should move to the field RES2. This is formulated as (note the sequence for execution):

```

res1
  COMMENT LEGAL USE label_result SHOW
  MUSTENTER
  TYPE COMMENT

```

```

AFTER ENTRY
  IF res1<>6 THEN
    res1sc=0
    GOTO res2
  ENDIF
END
END

```

Note:

1. The additional command GOTO followed by the field name. GOTO can be used alternatively to JUMPS and we generally give it preference. It would actually not be that simple to formulate an efficient JUMPS command in this case. Try to use JUMPS command to set this up as an exercise.
2. When you have connecting conditions (two or more conditions connected by AND or OR), then it will be mandatory to use parentheses. More about this later.

If you are in the field RES3, there is only the field RES3SC to follow, thus the above GOTO command directing to another following field becomes inappropriate. EpiData has a command GOTO WRITE. After this command, the user gets the prompting whether the record should be written to disk. We can thus use:

```

res3
  COMMENT LEGAL USE label_result SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    IF res3<>6 THEN
      res3sc=0
      GOTO WRITE
    ENDIF
  END
END

```

There is one final thing to be done. As has been mentioned in the data documentation sheet, if the value of the res1 is 6 (scanty, quantified), then it should not be possible to enter 0 (not applicable) in res1sc for obvious reasons. The same is the case for res2 and res3. If this happens, then a message should appear, "Values of res1 and res1sc are not compatible. Please verify" and the cursor should go back to res1. How do we achieve this? EpiData has a command HELP (which can be used to deliver messages to the person while doing data entry) after which one can write anything in quotation marks:

```

res1sc
  COMMENT LEGAL USE label_scanty SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    IF (res1=6) AND (res1sc=0) THEN
      HELP "Values of res1 and res1sc not compatible. Please verify"
      GOTO res1
    ENDIF
  END
END

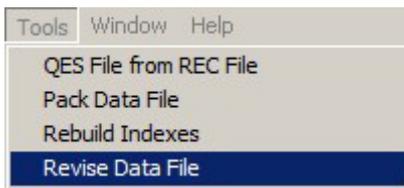
```

Note: Check commands added here can only be executed if and only if the cursor is in the field in question. You can circumvent Check file commands for a field if you use the

mouse to skip a few fields: the mouse should thus not be used for data entry. Actually, there is really no need for the mouse during data entry as one proceeds – and must proceed – from one field to the next which happens automatically or by confirming an entry with the Enter key. Nevertheless, people cannot be prevented from using the mouse and thus circumventing your carefully crafted Check file. For key variables, we will thus show in a later exercise how some control can still be retained.

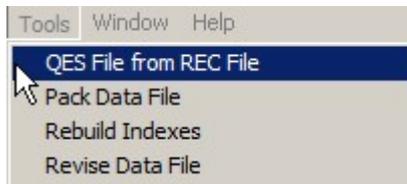
Now that you have learnt how to create QES, REC and CHK files, let us pause and reinforce some of the important principles of the EpiData triplet files.

1. The data structure is defined in the QES file. So, if you want to alter the structure of the database, that is, if **you want to add or delete fields**, change the type of the field, increase/decrease the length of the field, you have to do it in the QES file.
2. **Any change you make to the QES file requires that you update the REC file** with the changes you have made to the structure of the database. Many a time, you will forget this step and expect to see the revised REC file. Note the changes in the QES file are not automatically updated to the REC file – you have to update it. The usual way you have learnt to create a REC file, (**ALT+2** and clicking ‘Make data file’) though gets you an updated REC file, it will erase/overwrite the previously created file with an empty REC file. Hence, if there was any data entered previously, it will be lost. There is another way of revising the REC file without losing any data. This can be accomplished through ‘Tools-Revise Data File’.



3. An even simpler method is just to open the REC file with 4. Enter Data: whenever you do that, EpiData checks whether there is a QES file of the same name in the same folder, and if there is, whether the date of the QES file is older or newer than that of the REC file. If it is newer, then obviously the QES file has been revised after last data entry. Then EpiData prompts you asking whether you want to adapt the REC file to the revised QES file. If you had made a non-lossy change (adding a field, making a field definition longer, or trivial changes like correcting a Field label) you can safely confirm. Should you have made a lossy change (removing a field, shortening a field definition), EpiData will alert you that you are about to lose data and where you will lose them.
4. Data entry is done in the REC file. It is the REC file which contains all the data.
5. Every time you add/delete the checks in the CHK file, check the functionality by entering some dummy data into the REC file to ensure that checks are actually functional.
6. The names of the QES, REC and CHK files have to be the **same** for them to be linked and work properly. You now realize the value of making the file extensions visible!
7. **Always place the triplet files together in the same folder.** Whenever you have to share the files, share all the three together. Since the QES file is integrated into the REC file, REC and CHK files are enough for functional data entry. However, if you

have to change the structure of the database, then you will need the QES file. Though there is an option to create a QES from the REC file, (look under Tools – QES file from REC file) it is a good practice to keep all the three files together, not least because a QES file created from a REC file may lose some precious formatting beauty.



Familiarize yourself with opening, editing and closing the three files.

Tasks:

- o Open the existing A_EX02.QES file and complete it.*
- o Create the A_EX02.REC file (overwrite the existing one).*
- o Edit the A_EX02.CHK file, make all fields MUSTENTER fields*
- o Make Range and legal and label blocks where appropriate, but not both for the same variable.*
- o Edit the field to ensure that label blocks are shown and add other key commands as appropriate such as AFTER ENTRY with IF ... THEN ... ENDIF and HELP statements.*

Solution to Exercise 2: The QES-REC-CHK triplet

Key Point(s):

- For text fields it is frequently better to make them 'upper-case text' as 'm' is not the same as 'M', but even better is to avoid text fields altogether and to give preference to numeric coding with metadata in label blocks
- In the CHK file, it is always better to make all fields 'Must enter', except for automatically calculated variables. Then you will not have any missing values for the variable.

Tasks:

- o *Open the existing A_EX02.QES file and complete it.*
- o *Create the A_EX02.REC file (overwrite the existing one).*
- o *Edit the A_EX02.CHK file, make all fields MUSTENTER fields*
- o *Make Range and legal and label blocks where appropriate, but not both for the same variable.*
- o *Edit the field to ensure that label blocks are shown and add other key commands as appropriate such as AFTER ENTRY with IF ... THEN ... ENDIF statements.*

Solution:

The questionnaire would look as follows:

This is the questionnaire for the laboratory register

```
serno      Laboratory serial number #####   Enter 9001, 9002, ... if serial number
is not unique and write data entry note (F5)
regdate    Registration date <dd/mm/yyyy>   Enter 01/01/1800 if missing
sex        Examinee's sex #
age        Examinee's age in years ###     Enter 999 if missing
reason     Examination reason #
res1       Result of specimen 1 #
res1sc    Result of specimen 1 scanty #
res2       Result of specimen 2 #
res2sc    Result of specimen 2 scanty #
res3       Result of specimen 3 #
res3sc    Result of specimen 3 scanty #
```

EpiData Entry software is not case-sensitive, but the definition of values for text fields obviously is. Nevertheless, you make it best a habit to use lower-case for field names as some statistical packages are case-sensitive when it comes to field names.

or, if shown with “Preview data form”:

This is the questionnaire for the laboratory register

serno	Laboratory serial number	<input type="text"/>	Enter 9001, 9002, ... if serial number is not unique and write data entry note (F5)
regdate	Registration date	<input type="text"/>	Enter 01/01/1800 if missing
sex	Examinee's sex	<input type="text"/>	
age	Examinee's age in years	<input type="text"/>	Enter 999 if missing
reason	Examination reason	<input type="text"/>	
res1	Result of specimen 1	<input type="text"/>	
res1sc	Result of specimen 1 scanty	<input type="text"/>	
res2	Result of specimen 2	<input type="text"/>	
res2sc	Result of specimen 2 scanty	<input type="text"/>	
res3	Result of specimen 3	<input type="text"/>	
res3sc	Result of specimen 3 scanty	<input type="text"/>	

The *.REC file we created can be looked at in a text editor (like NotePad™ that comes with Windows™ or the powerful free text editor Crimson that is found on the course CD-ROM) and we see the following:

```
16 1 VLAB Filelabel: Exercise 2, Part A
_label1      1  1  30  0  0  0  0 112 This is the questionnaire for the laboratory register
#serno       1  3  30  37  3  0  4 112 serno      Laboratory serial number
_label2      41  3  30  0  0  0  0 112      Enter 9001, 9002, ... if serial number is not unique and write data entry not
_label3      121 3  30  0  0  0  0 112 e (F5)
_regdate     1  4  30  37  4  11 10 112 regdate     Registration date
_label4      49  4  30  0  0  0  0 112      Enter 01/01/1800 if missing
#sex         1  5  30  37  5  0  1 112 sex         Examinee's sex
#age         1  6  30  37  6  0  3 112 age         Examinee's age in years
_label5      40  6  30  0  0  0  0 112      Enter 999 if missing
#reason      1  7  30  37  7  0  1 112 reason      Examination reason
#res1        1  8  30  37  8  0  1 112 res1        Result of specimen 1
#res1sc      1  9  30  37  9  0  1 112 res1sc      Result of specimen 1 scanty
#res2        1  10 30  37  10 0  1 112 res2        Result of specimen 2
#res2sc      1  11 30  37  11 0  1 112 res2sc      Result of specimen 2 scanty
#res3        1  12 30  37  12 0  1 112 res3        Result of specimen 3
#res3sc      1  13 30  37  13 0  1 112 res3sc      Result of specimen 3 scanty
```

While this is perhaps not very informative to you at this point in time, you may note the simplicity of it. Have you ever tried to look at a spreadsheet in a text editor? You cannot, as it will not load and all you see is some gibberish. In contrast, this is a straight simple text file and its file size is just 1,382 bytes. In comparison, a Microsoft Word® 1997-2003 file containing the single letter “a”, nothing else, weighs in at 24,576 bytes...

For the *.CHK file, the entering of ranges and legal values took perhaps a bit trial and error. But basically it is very simple. For the field REGDATE we just entered:

```
01/01/2000-31/12/2005,01/01/1800
```

and for AGE

```
0-125,999
```

We can open the A_EX02.CHK file (CTRL+O, “Files of type”, “EpiData check file (*.chk)”) it is just a text file after all. It looks as follows:

```
LABELBLOCK
  LABEL label_sex
    1 Female
    2 Male
    9 "Sex not recorded"
  END
  LABEL label_reason
```

```

0 Diagnosis
1 "Follow-up at 1 month"
2 "Follow-up at 2 months"
3 "Follow-up at 3 months"
4 "Follow-up at 4 months"
5 "Follow-up at 5 months"
6 "Follow-up at 6 months"
7 "Follow-up at 7 months or later"
8 "Follow-up, month not stated"
9 "Reason not recorded"
END
LABEL label_result
0 Negative
1 "1+ positive"
2 "2+ positive"
3 "3+ positive"
4 "Positive, not quantified"
5 "Scanty, not quantified"
6 "Scanty, quantified"
9 "Result not recorded"
END
LABEL label_scanty
0 "Not applicable"
1 "1 AFB per 100 OIF"
2 "2 AFB per 100 OIF"
3 "3 AFB per 100 OIF"
4 "4 AFB per 100 OIF"
5 "5 AFB per 100 OIF"
6 "6 AFB per 100 OIF"
7 "7 AFB per 100 OIF"
8 "8 AFB per 100 OIF"
9 "9 AFB per 100 OIF"
END
END

serno
KEY UNIQUE 1
MUSTENTER
END

regdate
RANGE 01/01/2000 31/12/2005
LEGAL
01/01/1800
END
MUSTENTER
END

sex
COMMENT LEGAL USE label_sex SHOW
MUSTENTER
TYPE COMMENT
END

age
RANGE 0 125
LEGAL
999
END
MUSTENTER
END

```

```

reason
  COMMENT LEGAL USE label_reason SHOW
  MUSTENTER
  TYPE COMMENT
END

res1
  COMMENT LEGAL USE label_result SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    IF res1<>6 THEN
      res1sc=0
      GOTO res2
    ENDIF
  END
END

res1sc
  COMMENT LEGAL USE label_scanty SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    IF (res1=6) AND (res1sc=0) THEN
      HELP "Values of res1 and res1sc not compatible. Please verify"
      GOTO res1
    ENDIF
  END
END

res2
  COMMENT LEGAL USE label_result SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    IF res2<>6 THEN
      res2sc=0
      GOTO res3
    ENDIF
  END
END

res2sc
  COMMENT LEGAL USE label_scanty SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    IF (res2=6) AND (res2sc=0) THEN
      HELP "Values of res2 and res2sc not compatible. Please verify"
      GOTO res2
    ENDIF
  END
END

res3
  COMMENT LEGAL USE label_result SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    IF res3<>6 THEN

```

```

        res3sc=0
        GOTO WRITE
    ENDIF
END
END

res3sc
COMMENT LEGAL USE label_scanty SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
    IF (res3=6) AND (res3sc=0) THEN
        HELP "Values of res3 and res3sc not compatible. Please verify"
        GOTO res3
    ENDIF
END
END

```

Note that the capitalization versus the use of lower-case letters is used here only for easier visualization of the program flow (see note above about EpiData not being case-sensitive).

We will learn very soon how to edit the *.CHK file directly to experience its tremendous power.

Exercise 3: Derived fields and Check file commands unrelated to a specific field

At the end of this exercise you should be able to:

- a. Create automatically calculated fields from one or more existing fields by editing the *.qes and *.chk files
- b. Opening the whole CHK file and make additions to it that cannot be done at the field level

Is the laboratory serial number really unique?

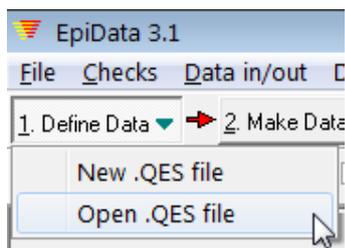
We noted earlier that the laboratory serial number is unique for one given year and for a single laboratory. If the study involves more than one year, then it is not unique; a combination of serial number and year of registration will make it unique. If the study involves more than one laboratory, then even this is not unique; you will then require a combination of laboratory identifier, serial number, year of registration to make it unique. How do we construct a variable which derives information from other variables entered and is auto-generated? That is the topic of our discussion now.

This is what we are going to do in this exercise. But first, we need to have a method (that we are going to use for every exercise from now on) to preserve what we already got and save it with a new name and then make the necessary changes.

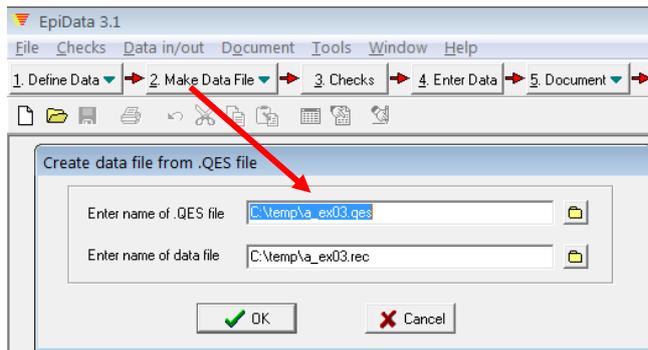
Making new *.QES, *.REC, and CHK files from existing files

First download the solution from Exercise 2 and overwrite your existing files. In this way, we reach a uniform place from where we can begin to build the next exercise. Then, let us use this to create the new set of EpiData triplets and name them A_EX03.

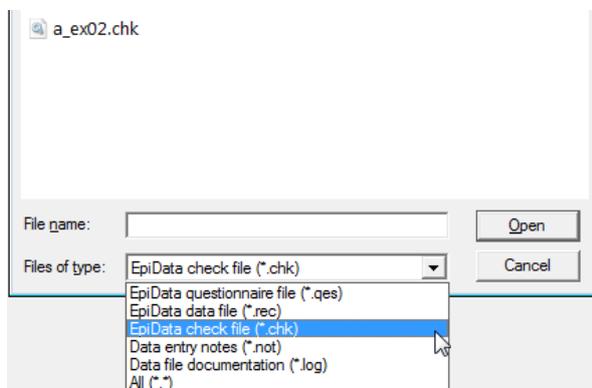
First, let us make A_EX03.QES from A_EX02.QES: This is easily done by opening:



the already existing A_EX02.QES and save it right away as A_EX03.QES. It is then edited as needed, saved and then the A_EX03.REC file is made from it as we did before:



What is missing is the A_EX03.CHK. Using “3. Checks” is not an option as this will create a new check file with none of the checks we have previously created. Hence, we need to open the A_EX02.CHK file and save it as the new A_EX03.CHK file. Click thus “File” | “Open” (or more simply **CTRL+O**) and select the correct file type:



Open A_EX02.CHK and save it as “A_EX03.CHK”. Before you close it, let’s remove the KEY UNIQUE from the field SERNO, as the latter will no more stay unique and IDCODE will become the new unique identifier:

```
serno
KEY UNIQUE 1
MUSTENTER
END
```

Save and exit.

Note: This is another way of opening the CHK file and here you will click on the CHK file to open it. Remember the other way of opening the CHK file - that is via “3. Checks”, you have to click on the REC file for which the checks have to be defined.

Task: We have to create a new variable IDCODE which will be auto-generated by the combination of laboratory identifier (let us say LABCODE), year of registration (which can be extracted from REGDATE) and laboratory serial number. This identifier will thus have a form like LABCODE-YEAR-SERNO.

As you will quickly notice, we will have to create three new fields, one for the unique identifier (IDCODE), one for laboratory identifier (LABCODE), and since it is easier to enter laboratory names rather than to remember codes for each laboratory before entry, we have to

create one more field for the laboratory name (LABNAME). Where do we have to go to create new fields? That's right! It's the QES file!!

Editing the new A_EX03.QES file

We will make one new field that is a MUSTENTER field, with the field name LABNAME. It is a text field (any string) and will have a length of 20 characters:

```
labname Name of the laboratory _____
```

Let us have a label block for the laboratory names and codes, where the laboratory name will be the Field value and the laboratory code will be the Value label. We will extract this laboratory code and make it a component of our derived variable.

Additionally we will make two fields that will be generated by the software, i.e. the CHK file will make the calculation and the result will be written into these two fields. Of course, if the value of a field is calculated by EpiData Entry, it is best not to allow a data entry person to get into such a field as this could lead to inadvertent deletion of the content. You will thus learn the use of the command NOENTER for such fields.

The first of these latter two fields will get the laboratory code: We have a label block for the laboratory names and codes, where the laboratory name is the Field value and the laboratory code is the Value label. Instead of writing (as we did in the previous exercise) the Value label to the right of the field definition, we will be writing it into another field. The field:

```
labcode Code of the laboratory _____
```

will also be a text field (any string) with the Field name LABCODE and a length of 4.

The second of the NOENTER fields will receive the new unique identifier, consisting of a combination of the laboratory code (length 4), the year of registration (length 4), and the laboratory serial number (length 4). This would give a length of 12, but for better visual display, we will add a hyphen between the three elements, thus 2 additional characters. The final length of this field with Field name IDCODE will thus be a string of 14 characters:

```
idcode Unique identifier _____
```

It doesn't really matter where NOENTER fields are placed, but for the sake of tidiness we prefer to place them separately from the MUSTENTER fields, for instance to the top of the questionnaire file:

This is the questionnaire for the laboratory register

```
labcode      Code of the laboratory  _____
idcode       Unique identifier    _____

labname      Name of the laboratory  _____
serno        Laboratory serial number #####  Enter 9001, 9002, ...
```

or, shown in the Data form preview:

This is the questionnaire for the laboratory register

labcode	Code of the laboratory	<input type="text"/>	
idcode	Unique identifier	<input type="text"/>	
labname	Name of the laboratory	<input type="text"/>	
serno	Laboratory serial number	<input type="text"/>	Enter 9001, 9002, ... i
regdate	Registration date	<input type="text"/>	Enter 01/01/1800
sex	Examinee's sex	<input type="text"/>	
age	Examinee's age in years	<input type="text"/>	Enter 999 if missing

Once done, save and make a new data (A_EX03.REC) file. *Note this important step of updating the REC file once you have made changes in the QES file.*

Editing the A_EX03.CHK file

Using “3. Checks” allows making all field-specific checks and it has the advantage that when you edit the checks they are checked for grammar when you “Accept and close”. We will thus start with that, conscious that we cannot do all here. We will later need to exit this mode and make additional changes in the Check file as a whole. Finally, we then return again to here.

Thus open the Check file for A_EX03.REC and make the fields LABCODE and IDCODE NOENTER fields but the field LABNAME a MUSTENTER field. The command NOENTER cannot be picked from the list, it must be entered manually using “Edit”.

The field IDCODE will newly become the unique identifier, thus edit the field to expand to:

```
idcode
  KEY UNIQUE
  NOENTER
END
```

EpiData will automatically add “1” to become “KEY UNIQUE 1” as we have already removed it from SERNO.

Similarly make the field LABCODE also NOENTER. It will look like this.

```
labcode
  NOENTER
END
```

We will make a label block for the field LABNAME, but with a single entry as follows:

```
LABEL Label_labname
  Awuna "ML_J"
END
```

“Awuna” is thus the Field value and “ML_J” the Value label. This differs from what we did before when the value was always an Integer. In fact, EpiData allows the value to be any of the following, an extraordinarily powerful feature of the software:

- An integer
- A float

→ A text

As before, we Edit the field but modify slightly. If doing as before, we would write:

```
labname
  COMMENT LEGAL USE label_labname SHOW
  TYPE COMMENT
  MUSTENTER
END
```

to show the label block as a pop-up box and to type the Value label to the right of the field. This time we do not want to type the Value label to the right of the field but to write the Value label (i.e. the laboratory code) into the field LABCODE that we made a NOENTER field. To this end, we expand the above and very simply add the name of the field after TYPE COMMENT:

```
labname
  COMMENT LEGAL USE label_labname SHOW
  TYPE COMMENT labcode
  MUSTENTER
END
```

After this, you have to write a command to generate (calculate) the IDCODE. For this we will use the AFTER ENTRY command. First, remember that we can generate IDCODE only after all the component variables have been entered. So, since the last of the component fields to be entered is REGDATE, we will write an AFTER ENTRY command for this variable.

```
regdate
  RANGE 01/01/2000 31/12/2005
  LEGAL
    01/01/1800
  END
  MUSTENTER
  AFTER ENTRY
    idcode=labcode+"-"+YEAR(regdate)+"-"+serno
  END
END
```

Note here that we have been able to extract the year out of the variable REGDATE. We can extract any component from a date field:

```
Extract day:      DAY(datefield)
Extract month:    MONTH(datefield)
Extract year:     YEAR(datefield)
```

Note also, that the hyphen that is to separate the components should be placed into quotation marks.

Is it possible to bypass the fields though it is **MUSTENTER**?

Unfortunately, the answer is yes!

The fact is that you have to enter the field to which any checks are to be applied. Once you enter the field, it will be difficult to exit without entering a value. But if you by-pass the fields using the mouse, then it is possible to skip the fields and leave them blank. If a field is blank, it is always a problem especially if this happens to be your unique identifier variable.

Is there a way to ensure that certain key fields cannot remain blank? Is there a way to circumvent this problem and ensure that no record can be saved without the unique identifier?

Fortunately, the answer is yes again!

However, to achieve this, we will need to manipulate the CHK file in a totally different way than we have been doing until now. Since this is a CHK command not specific to any field but related to the record as a whole, we cannot use the earlier method of editing the CHK file (Clicking on 3. Checks) which works strictly on the field level. We will have to open the full CHK file.

How do we do it?

With “File” | “Open” (or with **CTRL+O**) and click on the A_EX03.CHK file. Let us spend some time examining this file.

At its beginning, you see that EpiData placed there all Label blocks in one large LABELBLOCK that finishes with END:

```
LABELBLOCK
  LABEL label_labname
    Awuna ML_J
  END
  LABEL label_sex
    1 Female
    2 Male
    9 "Sex not recorded"
  END
  LABEL label_reason
    0 Diagnosis
    1 "Follow-up at 1 month"
    2 "Follow-up at 2 months"
    3 "Follow-up at 3 months"
    4 "Follow-up at 4 months"
    . . . .
  END
END
```

Each LABEL command inside also finishes with an END as we already saw when we wrote the Field labels.

Afterwards, the fields with their checks follow:

```
labcode
  NOENTER
END
```

Preventing that a record can be saved without an identifier

We have already used AFTER ENTRY with a field, there is also a BEFORE ENTRY that goes with a specific field. But then there are commands that are not related to a specific field:

```
BEFORE FILE
AFTER FILE
BEFORE RECORD
AFTER RECORD
```

These commands must therefore be written by opening the entire CHK file as demonstrated above. It doesn't really matter whether you put them before the fields or at the end of the record (but it will not work if you place them between fields). EpiData will rearrange things

and place them after the END command following the LABELBLOCK and before the first field. We might thus as well place them there ourselves, because then we make no placement error and see everything that is outside the field level just before the information at the field levels begins. *All the above commands are block commands and must finish with an END.*

What we had in mind was a command that would check after all information has been entered and the record is about to be saved, whether the unique identifier contains a value. If not, the entry person should be given a warning and then experience a forced return to the field that contains the first element contributing to the unique identifier. What we need is an AFTER RECORD statement:

```
AFTER RECORD
  Some commands to execute
END
```

These “Some commands ...” must check whether the field IDCODE is empty. The designation for “empty” is a period (.), thus:

```
AFTER RECORD
  IF idcode=. THEN
    Some commands to execute
  ENDIF
END
```

We use the command HELP to convey the relevant message to the person entering the data. At the line end, we can expand to indicate the type of pop-up box that should appear for the data entry person with an additional command:

```
AFTER RECORD
  IF idcode=. THEN
    HELP "There is no identifier" TYPE=WARNING
  ENDIF
END
```

So far, we get the warning displayed but if the user accepts it, nothing is going to happen and the record can still be saved to disk, which is, precisely what we want to prevent. We must therefore direct the cursor to go to the very first MUSTENTER field that contributes to the idcode:

```
AFTER RECORD
  IF idcode=. THEN
    HELP "There is no identifier" TYPE=WARNING
    GOTO labname
  ENDIF
END
```

This way it can be prevented that the data entry person ever gets out of the record without an identifier. That the identifier is unique is checked in the field itself and is of no concern here.

We can use the block commands mentioned above to interact with the data entry operator. For example we can provide a welcome message when the data entry operator opens the REC file by using a command BEFORE FILE or alert the person to take back-up of data when he finishes data entry for the day and closes the REC file using AFTER FILE. Observe the couple of examples below:

```
BEFORE FILE
  HELP "Welcome! Please do not use MOUSE"
END
```

AFTER FILE

```
HELP "Thank you! Remember to back-up your data"
```

END

Please note that all the block commands must finish with an END. Not mentioning anything with TYPE will provide the message as INFORMATION. The other options with TYPE are CONFIRMATION and ERROR.

Let us now turn to another concept.

Our identifier field IDCODE has a length 14 and is of the type LABCODE-YEAR-SERNO. However, because our serial number is an integer, its actual length might vary from 11 to 14 characters: the length of our IDCODE will not be uniform. Though, this will not affect functionality in any manner, it is cosmetically not very appealing. It is actually more than just visual: in analysis it becomes impossible to verify that each identifier is correctly coded with 14 position by looking at position 14. It will also cause issues with sorting which will no more follow sequential numbering, etc, etc.

Is it possible to write leading zeros to get a consistent length of 4 for SERNO and thus a consistent length of 14 for IDCODE?

It is possible if the field is a text field and thus can take what we call “leading zeros” like in “0023”. What we thus need to do is to create a temporary string variable of length 4 into which the serial number is written, with leading zeros as appropriate, and be kept until used to construct the value for the field IDCODE.

What are these temporary variables? How to create them?

Here, we are introducing a new, powerful feature of EpiData, called “temporary variables”. What are these? These variables are defined in the CHK file and not in the QES file (as we have been doing until now for all variables). Hence, they will be retained in the memory but will not become part of the actual dataset. This can be very useful, if some intermediate calculations are to be made and these should be staying in memory until they are used. Temporary variables can have a field name length of 16 (rather than the usual 10). This is of very practical utility, because we can just add then the suffix “Temp” to any “hard-wired” variable, recognize it immediately, and never have to worry that it is too long.

The creation of this temporary variable would go into a BEFORE FILE statement and be globally valid. The place for such a statement is also after the LABELBLOCK ... END. The command is written as:

```
BEFORE FILE
  DEFINE sernoTemp ____ GLOBAL
END
```

Note: The optional term GLOBAL helps to inform EpiData that this can be used across the records and related databases. In this specific example, we could omit it and we could also use BEFORE RECORD instead of BEFORE FILE.

With this, we are done in the open Check file. While we could finalize everything here, it is better to do things that work at field level with “3. Checks” as the grammar is being checked there.

Editing more Checks at the field level

Open Checks (**ALT+3**) for A_EX03.REC to edit Checks at the field level. The first thing to tackle is to write the SERNO into the SERNOTEMP string field, with leading zeros where indicated. This must be done in an AFTER ENTRY statement in the field SERNO which we expand accordingly:

```
serno
  MUSTENTER
  AFTER ENTRY
    Do something
  END
END
```

EpiData executes commands sequentially line by line. A good approach is thus to assign a default value to a variable and to define subsequently and sequentially, respecting the hierarchy, the conditions for which exceptions must be made. Thus, as a default, we could assign that the temporary variable takes the same value as SERNO:

```
serno
  MUSTENTER
  AFTER ENTRY
    sernoTemp=serno
    Do something
  END
END
```

The flow of the hierarchy suggests that the next change to make is if the SERNO is less than 1,000, then if it is less than 100, and finally if it is less than 10. If it is less than 1,000 (but 100 or more), only one leading zero has to be added:

```
serno
  MUSTENTER
  AFTER ENTRY
    sernoTemp=serno
    IF serno<1000 THEN
      sernoTemp="0"+serno
    ENDIF
    IF ...
      Do the hierarchically next thing
    ENDIF
    ...
  END
END
```

Note that “0” is in quotation marks because it is text, not a number. It is connected with a plus to a number but because the field is text and the first character (0) is text, this will result in a string. If the value of SERNO were 511, then the value of SERNOTEMP would be “0511”. You will have an opportunity during task solving to complete this block for yourself. In order to prevent an error message, save it for the time being with only the first ‘IF’ statement. Also, IDCODE has to be created using sernoTemp rather than serno.

Tasks:

- o Complete the revised A_EX03.CHK file and verify your QES-REC-CHK triplet
- o Test the system by trying to enter data (4. Enter data)

Solution to Exercise 3: Derived fields and Check file commands unrelated to a specific field

Key Point(s):

- Commands that are specific for a field should be written by accessing the 3. Checks tab in the process bar as there they are checked for the accuracy of the grammar
- For commands that pertain to the file or the record rather than a field, open the Check file to make the necessary modifications. Note that the grammar is not checked here (errors will be reported when trying to open the data file for data entry).

Tasks:

- o Complete the revised A_EX03 .CHK file and verify your QES-REC-CHK triplet
- o Test the system by trying to enter data (4. Enter data)

Solution:

This is the Questionnaire file A_EX03 .QES:

This is the questionnaire for the laboratory register

labcode	Code of the laboratory	<input type="text"/>	
idcode	Unique identifier	<input type="text"/>	
labname	Name of the laboratory	<input type="text"/>	
serno	Laboratory serial number	<input type="text"/>	Enter 9001, 9002, ... if serial number is not unique and write data entry note (F5)
regdate	Registration date	<input type="text"/>	Enter 01/01/1800 if missing
sex	Examinee's sex	<input type="text"/>	
age	Examinee's age in years	<input type="text"/>	Enter 999 if missing
reason	Examination reason	<input type="text"/>	
res1	Result of specimen 1	<input type="text"/>	
res1sc	Result of specimen 1 scanty	<input type="text"/>	
res2	Result of specimen 2	<input type="text"/>	
res2sc	Result of specimen 2 scanty	<input type="text"/>	
res3	Result of specimen 3	<input type="text"/>	
res3sc	Result of specimen 3 scanty	<input type="text"/>	

This is the Check file A_EX03 .CHK:

```
LABELBLOCK
  LABEL label_labname
    Awuna ML_J
  END
  LABEL label_sex
    1 Female
    2 Male
```

```

    9 "Sex not recorded"
END
LABEL label_reason
    0 Diagnosis
    1 "Follow-up at 1 month"
    2 "Follow-up at 2 months"
    3 "Follow-up at 3 months"
    4 "Follow-up at 4 months"
    5 "Follow-up at 5 months"
    6 "Follow-up at 6 months"
    7 "Follow-up at 7 months or later"
    8 "Follow-up, month not stated"
    9 "Reason not recorded"
END
LABEL label_result
    0 Negative
    1 "1+ positive"
    2 "2+ positive"
    3 "3+ positive"
    4 "Positive, not quantified"
    5 "Scanty, not quantified"
    6 "Scanty, quantified"
    9 "Result not recorded"
END
LABEL label_scanty
    0 "Not applicable"
    1 "1 AFB per 100 OIF"
    2 "2 AFB per 100 OIF"
    3 "3 AFB per 100 OIF"
    4 "4 AFB per 100 OIF"
    5 "5 AFB per 100 OIF"
    6 "6 AFB per 100 OIF"
    7 "7 AFB per 100 OIF"
    8 "8 AFB per 100 OIF"
    9 "9 AFB per 100 OIF"
END
END

BEFORE FILE
    DEFINE sernoTemp ____ GLOBAL
END

AFTER RECORD
    IF idcode=. THEN
        HELP "You cannot save a record without an identifier\n Please enter
all required information" TYPE=WARNING
        GOTO labname
    ENDIF
END

labcode
    NOENTER
END

idcode
    KEY UNIQUE 1
    NOENTER
END

labname
    COMMENT LEGAL USE label_labname SHOW

```

```

MUSTENTER
REPEAT
TYPE COMMENT labcode
END

serno
MUSTENTER
AFTER ENTRY
    sernoTemp=serno
    IF serno<1000 THEN
        sernoTemp="0"+serno
    ENDIF
    IF serno<100 THEN
        sernoTemp="00"+serno
    ENDIF
    IF serno<10 THEN
        sernoTemp="000"+serno
    ENDIF
END
END

regdate
RANGE 01/01/2000 31/12/2005
LEGAL
    01/01/1800
END
MUSTENTER
AFTER ENTRY
    idcode=labcode+"-"+year(regdate)+"-"+sernoTemp
END
END

sex
COMMENT LEGAL USE label_sex SHOW
MUSTENTER
TYPE COMMENT
END

age
RANGE 0 125
LEGAL
    999
END
MUSTENTER
END

reason
COMMENT LEGAL USE label_reason SHOW
MUSTENTER
TYPE COMMENT
END

res1
COMMENT LEGAL USE label_result SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
    IF res1<>6 THEN
        res1sc=0
        GOTO res2
    ENDIF

```

```

END
END

res1sc
COMMENT LEGAL USE label_scanty SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
    IF (res1=6) AND (res1sc=0) THEN
        HELP "Values of res1 and res1sc not compatible. Please verify"
        GOTO res1
    ENDIF
END
END

res2
COMMENT LEGAL USE label_result SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
    IF res2<>6 THEN
        res2sc=0
        GOTO res3
    ENDIF
END
END

res2sc
COMMENT LEGAL USE label_scanty SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
    IF (res2=6) AND (res2sc=0) THEN
        HELP "Values of res2 and res2sc not compatible. Please verify"
        GOTO res2
    ENDIF
END
END

res3
COMMENT LEGAL USE label_result SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
    IF res3<>6 THEN
        res3sc=0
        GOTO WRITE
    ENDIF
END
END

res3sc
COMMENT LEGAL USE label_scanty SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
    IF (res3=6) AND (res3sc=0) THEN
        HELP "Values of res3 and res3sc not compatible. Please verify"
        GOTO res3
    ENDIF
END
END

```

END

This is a completed data entry form:

This is the questionnaire for the laboratory register

labcode	Code of the laboratory	<input type="text" value="ML_J"/>
idcode	Unique identifier	<input type="text" value="ML_J-2002-0023"/>
labname	Name of the laboratory	<input type="text" value="Awuna"/>
serno	Laboratory serial number	<input type="text" value="23"/> Enter 9001, 9002, ... if serial number is not unique and write data entry note (F5)
regdate	Registration date	<input type="text" value="20/03/2002"/> Enter 01/01/1800 if missing
sex	Examinee's sex	<input type="text" value="1"/> Female
age	Examinee's age in years	<input type="text" value="23"/> Enter 999 if missing
reason	Examination reason	<input type="text" value="0"/> Diagnosis
res1	Result of specimen 1	<input type="text" value="6"/> Scanty, quantified
res1sc	Result of specimen 1 scanty	<input type="text" value="4"/> 4 AFB per 100 OIF
res2	Result of specimen 2	<input type="text" value="9"/> Result not recorded
res2sc	Result of specimen 2 scanty	<input type="text" value="0"/>
res3	Result of specimen 3	<input type="text" value="9"/> Result not recorded
res3sc	Result of specimen 3 scanty	<input type="text" value="0"/> Not applicable

Exercise 4: Data entry and validation

At the end of this exercise you should be able to:

- Know the three ways of reducing data entry errors
- Copy the structure of a REC file
- Export data from EpiData files
- Validate duplicate data files

You have a line listing of 15 records on the page following the task description. These data should be entered in this exercise. But before you start working, a few considerations are in place.

Ensuring quality data entry

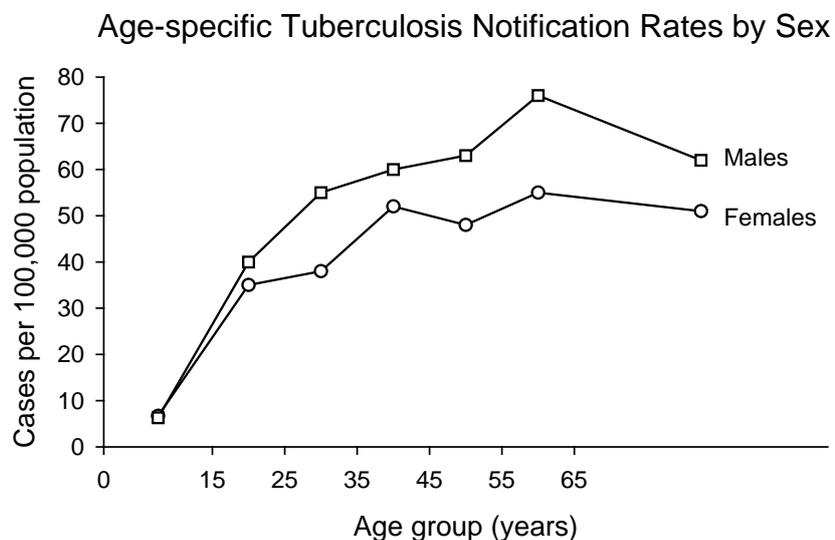
The motto for this course is:

“You wish never to find yourself in a position to defend the quality of your data”

Michael B Gregg, formerly MMWR Editor, deceased

You might be challenged about the interpretation of your data, that is part of the scientific process, but your data should be of impeccable quality.

What do you think about the following graph?



It looks nice and we could talk about the differences between males and females and this and that. But we will keep it short: it is total nonsense. The data underlying this graph have no basis, they were made up. Of course, if we were to present these data for real, it would be outright scientific fraud. Few people commit that (but it exists). *Nevertheless, often no assurance can be given that the computerized data are a true reflection of the original data source.* People may have in all honesty done “their best” and assume that they made no errors or so few that it really doesn’t matter. However, this is not good enough for science in general and public health and epidemiology in particular.

There are three ways how we reduce and ultimately eliminate data entry errors:

- o Using a *.CHK file
- o Working together
- o Duplicate data entry and validation

Using a *.CHK file

We have already a few inbuilt conditions that limit data entry errors by creating the A_EX03.CHK file. For instance, a MUSTENTER field will prevent a data entry person to skip an actually recorded value, as one cannot continue without having entered a value for that field. For the field SEX, we allowed only 1, 2, and 9 as legal values. It is thus not possible to enter “3” into this field. Combined with the pop-up menu during entry, no confusion can arise. The *.CHK file is an extremely powerful tool to control how data entry can be controlled through restrictions.

Working together

Entering data alone requires continuously shifting attention between the paper record and the computer screen. This will almost by necessity result in numerous errors, be it that a record is skipped or that it is forgotten what we just read. It should be routine that two persons work on data entry: one person reads aloud the Field value, the other repeats it aloud and enters the value.

Duplicate data entry and validation

Even with both of the above precautionary measures, data entry errors will still occur, and worse, to an unknown extent. ***The only way, and the only acceptable one, is to enter the data twice into two different files, and then to compare the two files for discordances.*** Any discordance uncovered will then be corrected against the original paper record.

The rationale behind this process is: *the probability of committing the same error in the same field twice when data entry is done independently by two persons is very small.* Hence, if we list all the discrepancies by comparing the two databases and correct all of them, then we can be reassured that the remaining frequency data entry errors is miniscule.

EpiData Entry provides this powerful tool and it offers two approaches to it. The first approach is to enter the data independently twice. The second approach is to prepare for duplicate entry. After the first file is completed, the second file is prepared based on a key field for the first file. While then entering the second duplicate file, the value is checked for each field in each record against the same record of the first file while entering it and you are

warned of any discordance, so that you can ensure proper recording during the second entry process.

In either case, we need a unique identifier. We have made a provision that we have such an identifier (see previous exercises). Sometimes an identifier must be constructed from more than one variable as we have shown.

If a duplicate key is revealed (because there is a perhaps a problem with a component contributor), then a data entry note (a *.not file) should be written. This can be invoked with **F5**. In this note, you must specify exactly with what identifier you have replaced the duplicate key, so that this note can be passed on to those who enter the data the second time, enabling them to use the same alternative key.

At this point in time, you will be using the first approach (we recommend this approach given its strength of enabling perfect documentation for future audits), and that is to independently enter the 15 records twice and then to compare the two files.

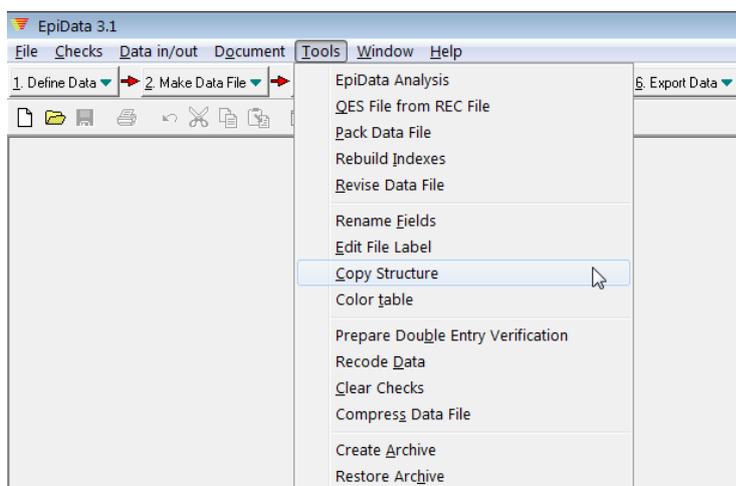
Note for data entry: Do never move around the fields with the help of the mouse. The mouse movements cannot be recorded properly and unforeseen errors may occur (e.g., bypassing a calculation made in a field, missing a MUSTENTER command, etc), because the Check file cannot be applied to fields you skip by moving the mouse from one to another. Use only TAB, cursor keys and the Enter key to move around an EpiData entry form.

Before you get to actually enter the data, you find here some assistance to make your data entry work more efficient.

Make a duplicate of the REC and CHK file pair

As we are entering the same data twice, we need two pairs of REC / CHK files, one for the first, the other for the second entry.

In “Tools” you find “Copy structure”:



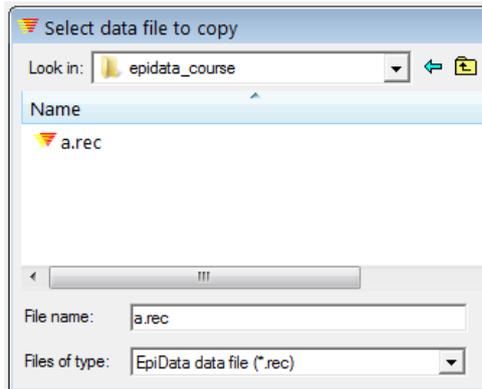
This allows making a copy of a REC file together with its CHK file to a new empty pair of files without copying the data. Thus, if you have A.REC and A.CHK and you use this feature,

you can copy both the REC file and the CHK in one sweep to an empty B.REC and B.CHK file pair.

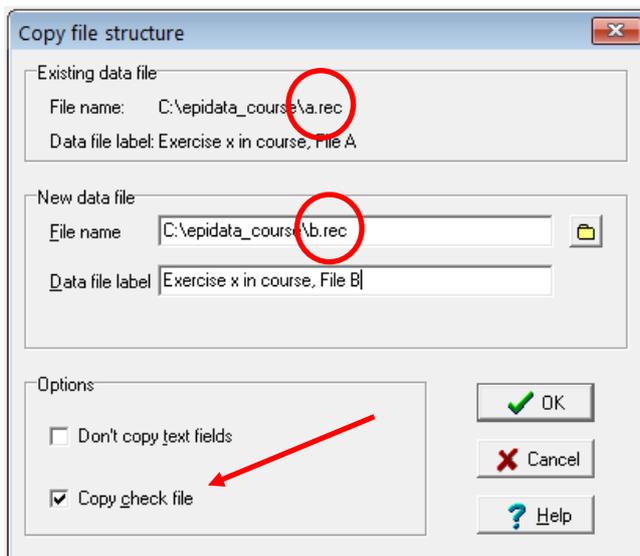
Tasks:

- o **Download the solution of Exercise 3 and overwrite your A_EX03 triplet files. Go to “Tools” “Copy structure” and copy the A_EX03.REC including its A_EX03.CHK file to:**

A_EX04 A.REC and A_EX04A.CHK files



After selecting the A.REC file you get to the menu that allows you to give the desired name (here B.REC) to the New data file and keep the default Copy check file ticked:



Both new files are created at the same time and you get confirmation when you enter:



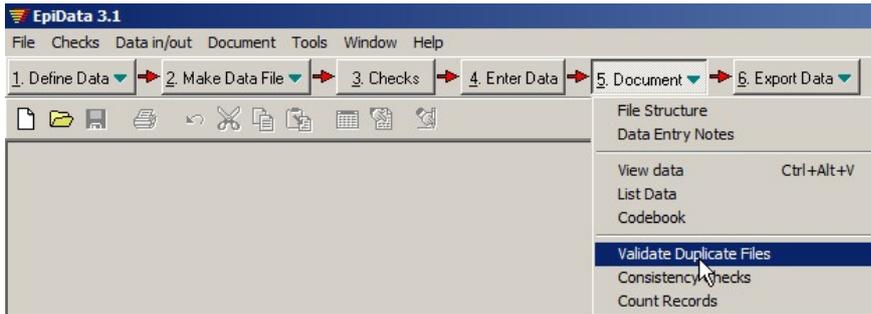
Again, note that this is for copying the structure (empty REC file and associated CHK file), not exporting data.

Double-entry

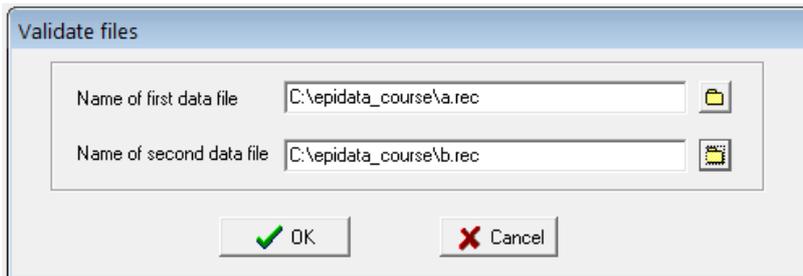
After you have both file pairs, enter the data into the A .REC (controlled by the A .CHK) and when completed, repeat data entry into the B .REC (controlled by the B .CHK).

Data validation

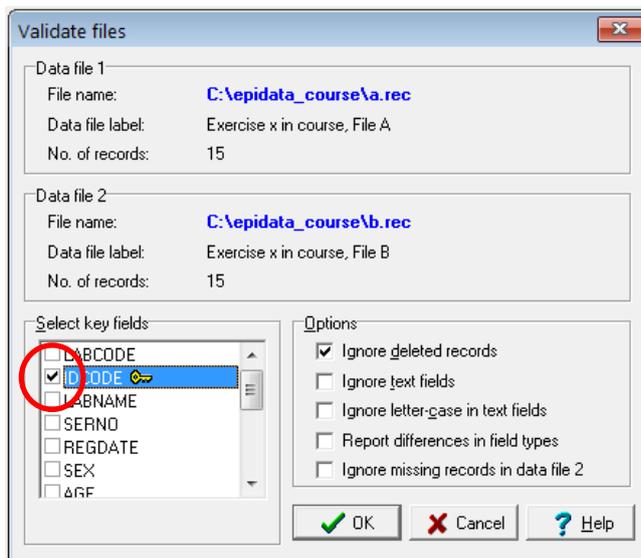
After completing double-entry, the two data files are compared, a procedure termed “**data validation**”. In the process bar, go to 5. Document and choose Validate Duplicate Files:



Enter the names of the two files that you need to compare into the respective two lines:



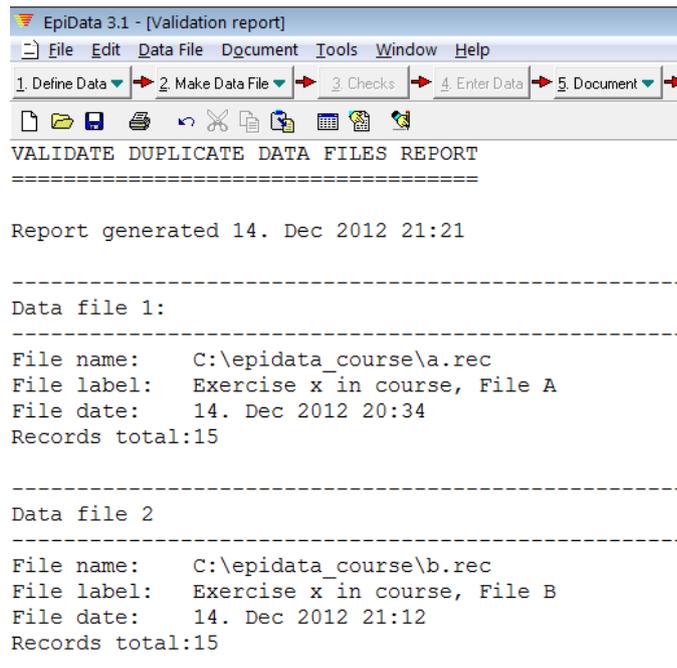
Confirming with OK opens:



We must compare the two files on “something”, and that something is our unique identifier IDCODE which we assured to be unique with KEY UNIQUE, and has therefore a little key to the right of the field name. We tick this field. This means that the IDCODE of a given record in File A determines that EpiData looks at the record with the same IDCODE in File B, irrespective of where in the sequence of records in the dataset it is found. This makes the sequence of entry of records irrelevant.

The Validation report

Once you OK, EpiData produces the Validation report:



```

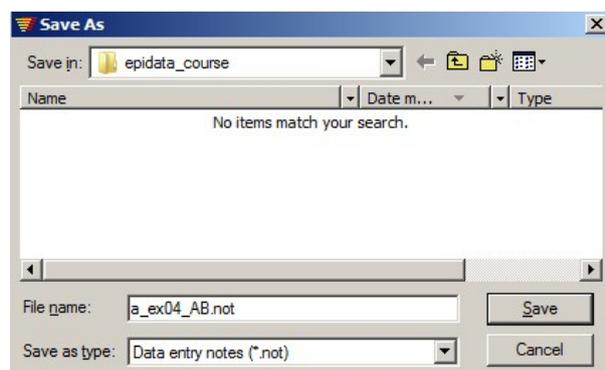
EpiData 3.1 - [Validation report]
File Edit Data File Document Tools Window Help
1. Define Data 2. Make Data File 3. Checks 4. Enter Data 5. Document
VALIDATE DUPLICATE DATA FILES REPORT
=====
Report generated 14. Dec 2012 21:21

-----
Data file 1:
-----
File name:      C:\epidata_course\a.rec
File label:     Exercise X in course, File A
File date:      14. Dec 2012 20:34
Records total: 15

-----
Data file 2
-----
File name:      C:\epidata_course\b.rec
File label:     Exercise X in course, File B
File date:      14. Dec 2012 21:12
Records total: 15
  
```

Verify first that 1) the number of records is the same in both files and 2) that nowhere a record is present in File A but missing in File B and vice versa. Should you find missing records, you must add them first before repeating validation. Once you have assured that both datasets have the same records, you must save the Validation report. This is not only essential for subsequent corrections (if needed), but to have a permanent record that you did indeed validate the two files and what the results were. This is of critical importance.

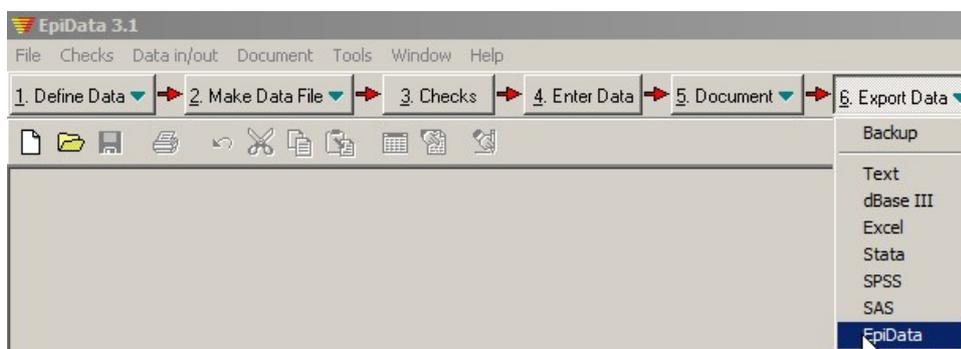
Validation files are simple text files with the extension *.NOT. Give it a name that makes it intuitively clear what this validation refers to, like *_AB.NOT:



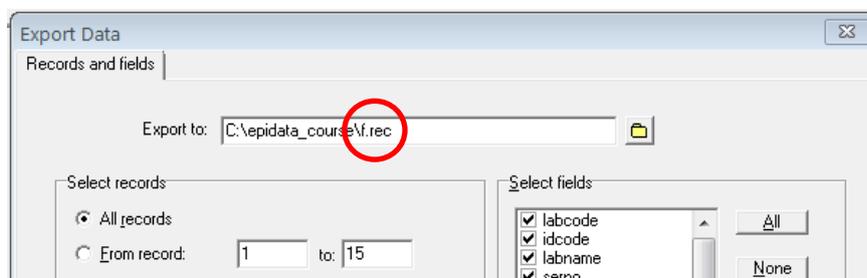
Creating a final dataset

One might be tempted to make corrections of any errors that might be identified through discordances in either the A.REC or the B.REC file. Doing so would, however, violate the “chain of evidence”: you could never repeat the validation process and get the same result, but data quality-assurance requires that the validation process is actually exactly reproducible. Therefore, the corrections must be made in a third file. To this end, we export the data from one of the source files to another EpiData file that we will call the F.REC file. To standardize as many things as possible, we always export the A.REC file to the F.REC file (even if in fact it is irrelevant whether we use the A.REC or the B.REC file, but consistency is good policy).

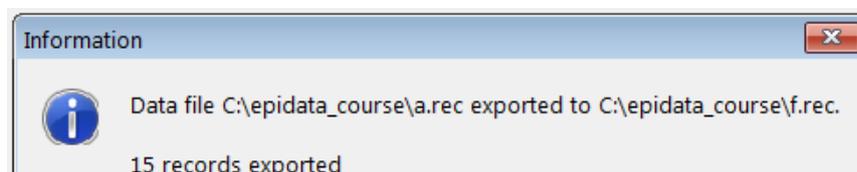
Go to “6. Export Data” | “Epidata”:



and export the A.REC file to the F.REC file:



You get confirmation:



Note: While this process of exporting data is perfectly fine for exporting the dataset that we created, we find that there are some errors (detailed in the next chapter) in the export of the CHK file. Unfortunately, this is due to **a bug in EpiData** during the export of the CHK file. The EpiData Association is conscious of these issues (which have been discovered only after work on the new EpiData Manager started in earnest). There are no plans to fix them because of the current work on the revision of EpiData software which are well on the way. The

EpiData Manager will combine the EpiData triplet of QES, REC and CHK files and have a different architecture altogether. Every effort will be made to ensure bug-free functionality in data export from the current version to the new version.

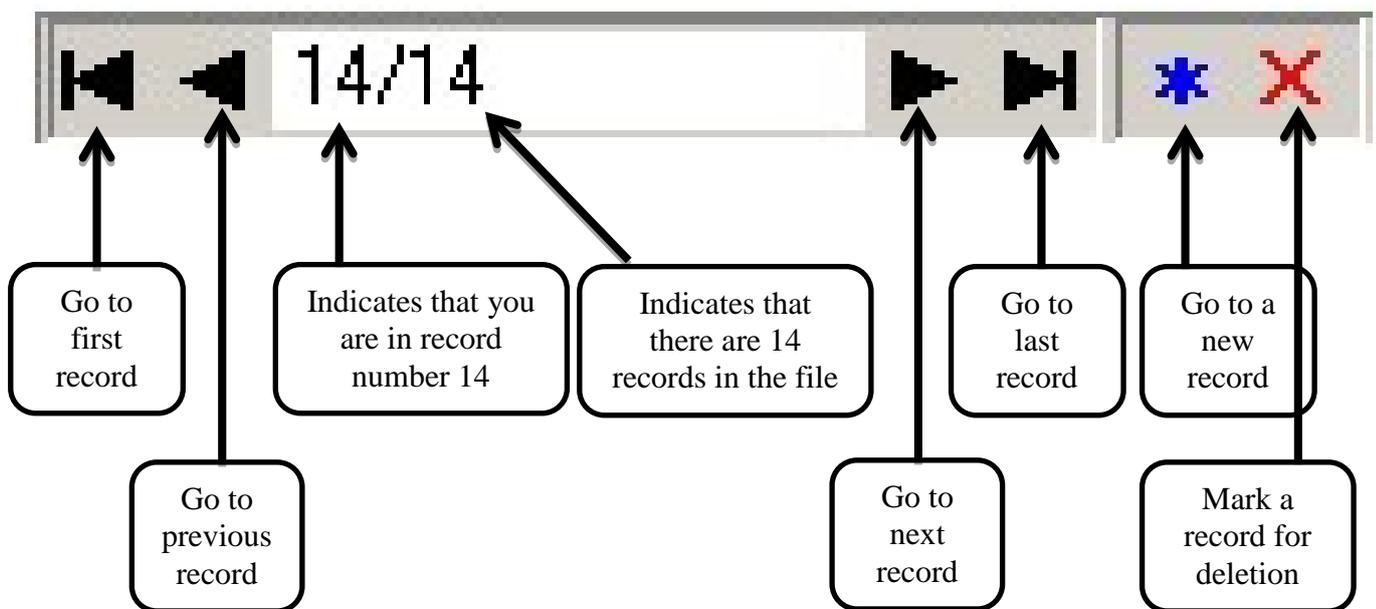
Hence, we recommend an alternative option as a workaround for the time being. Just create a copy of the REC and CHK files and rename them appropriately!

How to navigate through a REC file?

Before you begin, a few more tips on navigating through the REC file and manipulating them will help you in this exercise.

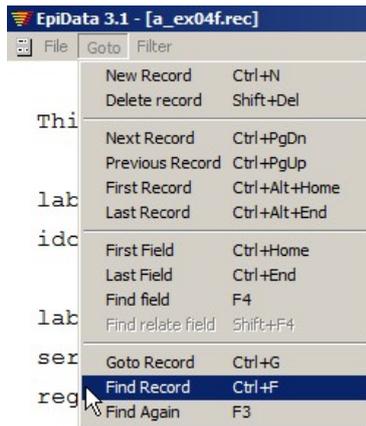
To navigate through different fields of a record, we recommend to use the ENTER key or ARROW keys or TAB key. Avoid using mouse since there is a possibility that checks are not applied when you use the mouse.

To navigate between the records of the REC file use the navigation panel on the left bottom end of the data entry screen which can be used to navigate through the records (see the diagram below to understand the different icons):

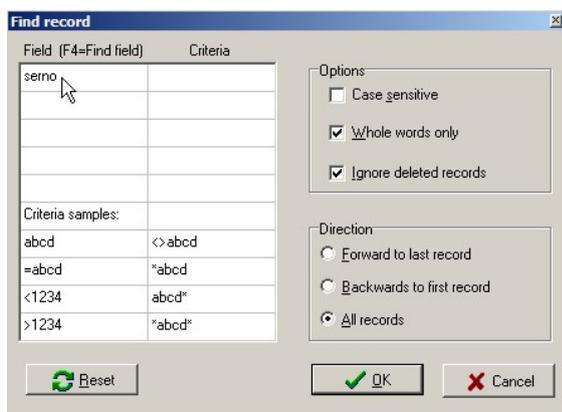


If you need to find a specific record and know the record number, use the function GOTO record (**CTRL+G**). If you do not know the record number, but want to find a record by specifying some criteria, use the function FIND record (**CTRL+F**). Let us demonstrate the latter option.

Let us say you want to find a record with the record with the unique identifier ML_J-2003-3303. Then click on GOTO in the menu and choose “Find Record”.

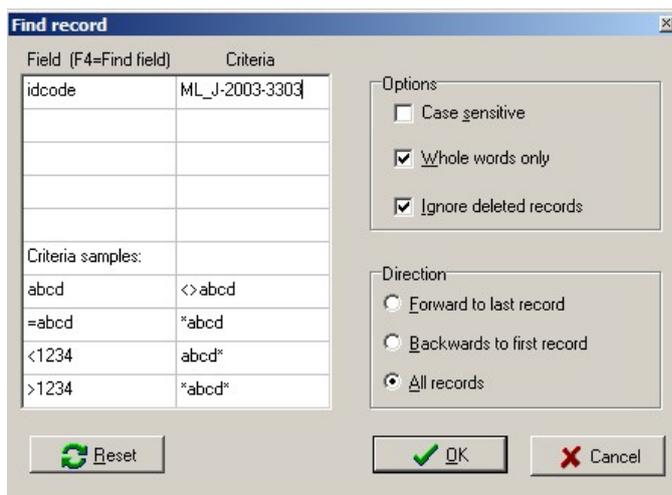


Click on it to find the dialogue box below



Note that you can get to the same place using the shortcut (CTRL+F).

Now, edit the Field and the Criteria as shown below:



Click OK and then you will find yourself in the record with the sought after identifier:

EpiData 3.1 - [a_ex04f.rec]
 File Goto Filter Window Help

This is the questionnaire for the laboratory register

labcode Code of the laboratory ML_J
 idcode Unique identifier ML_J-2003-3303

labname Name of the laboratory Awuna
 serno Laboratory serial number 3303 Enter 9001,
 regdate Registration date 26/10/2003 Enter

How to delete a record?

Deleting a record consists of two steps – first, marking a record for deletion; second, permanently deleting it. This is just a safety feature in EpiData to ensure the deletion of record does not happen by chance.

Steps in marking a record for deletion (Look at the screenshot below)

1. Open the REC file and go to the record you want to delete.
2. Click on the red 'cross' mark next to the navigation panel at the left bottom of the data entry screen. The word DEL appears at the side of the red 'cross' mark.
3. Click the arrow in the navigation panel to go to the next record. This will prompt you to save the record. Click 'Yes' and this successfully marks the record for deletion.
4. Note that the record is not yet permanently deleted from the database. If you realize that this record was not to be deleted, you can undo the action by clicking on the same button and saving the record. DEL will disappear now: the red "cross" is a toggle key:

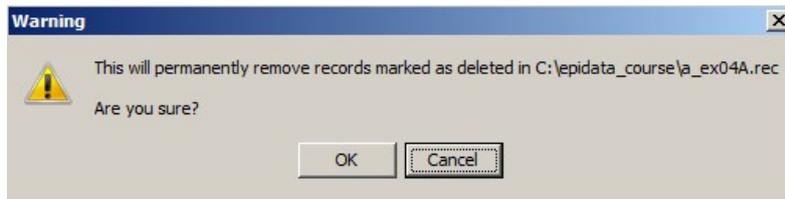


How to permanently delete a record? (Pack Data Files)

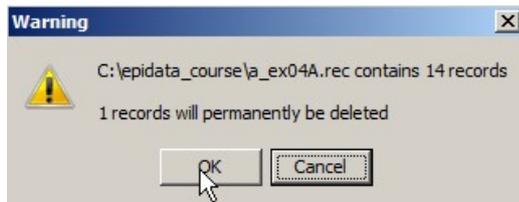
First go to Tools, select Pack Data File and click on it.



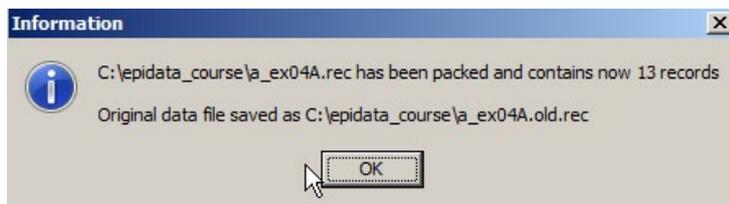
You will receive a warning about permanent deletion of the record:



If you press OK you will again receive another warning – this time more specifically mentioning how many records will be deleted. We hope you appreciate the effort of EpiData to protect your data!



If you click OK now, this will delete the records marked for deletion. But EpiData will preserve the original REC file as a_ex04A.old.rec in case you want to revert back!



You should now be prepared for the task of entering, double-entering, and validating the data

Tasks:

- o Download the solution of Exercise 3 and overwrite your A_EX03 triplet files. Go to “Tools” “Copy structure” and copy the A_EX03.REC including its A_EX03.CHK file to:
A_EX04 A.REC and A_EX04A.CHK files
A_EX04 B.REC and A_EX04B.CHK files*
- o Enter the 15 records using the A_EX04A.REC file. After completing data entry, enter the same data again into to the A_EX04B.REC file.*
- o After you have completed the two files, go to “5. Document” “Validate Duplicate Files” and produce a *.NOT file giving you a list of discordances, if any. Save the *.NOT file as A_EX04AB_validation.NOT*
- o Use “6. Export Data” “Epidata” to export either one of the two files [recommended: the A_EX04A.REC] to a new A_EX04F.REC file, and then make all corrections in this file. This is your final dataset.*

On the next page you find the dataset with 15 records

Laboratory: Awuna

Tuberculosis laboratory register

Year: 2003

Lab Serial No.	Date specimen received	Name	Sex M/F	Age	Name of referring facility	Address - patient for diagnosis	Reason for examination*		Results of specimen			Only for SS+ for diagnosis: TB Number or BMU**	Remarks
							Diagnosis (tick)	Month of follow up	1	2	3		
3298	26 Oct	Mary	F	35	Bindura	Beijingstr. 6		5	neg	neg			
3299	26 Oct	John	M	20	Awuna	Tokyo Ave 5	√		neg	neg	neg		
3300	26 Oct	Petra	F	30	Birchenough	Bangkok Rd 108		5	neg	neg			
3301	26 Oct	Charles	M	24	Bindura	Hanoi Street 7a		2	neg	neg			
3302	26 Oct	Tiffany	F	38	Bindura	Hongkong Ave 8	√		neg	neg	neg		
3303	26 Oct	George	M	60	Bindura	Zurich Rd 923	√		neg	neg	neg		
3304	26 Oct	Luke	M	78	Awuna	Paris Street 18a	√		neg	neg	neg		
3304	26 Oct	Virginia	F	28	Birchenough	London Rd 24	√		neg	neg	neg		
3305	27 Oct	David	M	50	Awuna	Baltimore Str 1		6	neg	neg			
3306	27 Oct	Hans	M	50	Ganda Chivua	Bern Str 12	√		1+	1+	1+	Ganda Chivua No 342	
3307	27 Oct	Bill	M	68	Bindura	Berlin Ave 88	√		neg	neg	neg		
3308	27 Oct	Susan	F	29	Birchenough	Amsterdam Rd 3		5	neg	neg			
3309	27 Oct	Marc	M	36	Bindura	Vienna Str 76		2	neg	neg			
3310	27 Oct	Eve	F	15	Awuna	Rome Ave 4		5	neg	neg			
3311	27 Oct	Anthony	M	37	Birchenough	Antwerp Str 26c		6	neg	neg			

* Check the appropriate category from the *Request for Sputum Examination*

**TB register number or name of the referral BMU (Basic Management Unit)

Solution to Exercise 4: Data entry and validation

Key Point(s):

- It should be routine that two persons work on data entry, and never one.
- The only and acceptable way to minimize data entry errors is to enter the data twice into two different files, and then compare the two files for discordances.
- Avoid using the mouse to move around fields during data entry, because the Check file cannot be applied to fields you skip by moving the mouse from one field to another.

Tasks:

- o **Download the solution of Exercise 3 and overwrite your A_EX03 triplet files. Go to “Tools” “Copy structure” and copy the A_EX03.REC including its A_EX03.CHK file to:**

A_EX04 A.REC **and** A_EX04A.CHK **files**

A_EX04 B.REC **and** A_EX04B.CHK **files**

- o **Enter the 15 records using the A_EX04A.REC file.**

After completing data entry, enter the same data again into to the A_EX04B.REC file.

- o **After you have completed the two files, go to “5. Document” “Validate Duplicate Files” and produce a *.NOT file giving you a list of discordances, if any. Save the *.NOT file as A_EX04AB_validation.NOT**
- o **Use “6. Export Data” “Epidata” to export either one of the two files [recommended: the A_EX04A.REC] to a new A_EX04F.REC file, and then make all corrections in this file. This is your final dataset.**

Solution:

Depending on the errors you made, you will get an output like the following:

```
VALIDATE DUPLICATE DATA FILES REPORT
=====
```

```
Report generated 19. Jun 2012 12:32
```

```
-----
Data file 1:
```

```
-----
File name:      C:\temp\a_ex04a.rec
File label:     Exercise 4, 1st entry, Part A
File date:      19. Jun 2012 12:23
Records total:15
```

```
-----
Data file 2
-----
```

File name: C:\temp\a_ex04b.rec
File label: Exercise 4, 2nd entry, Part A
File date: 19. Jun 2012 12:32
Records total:15

Options for validation:
Ignore deleted records: Yes
Ignore text fields: No
Ignore letter-case in text fields: No
Report differences in field types: No
Ignore missing records in data file 2 No

Fields in both data files that were used in the validation:
LABCODE, IDCODE, LABNAME, SERNO, REGDATE, SEX, AGE, REASON, RES1, RES1SC, RES2,
RES2SC, RES3, RES3SC

Fields excluded from data file 1:
None

Fields excluded from data file 2:
None

Fields used as index keys:
IDCODE

RESULTS OF VALIDATION:

Records missing in data file 1: 0
Records missing in data file 2: 0

Number of common records found: 15
Number of fields checked per record: 14
Total number of fields checked: 210

3 out of 15 records had errors (20.00 pct.)
3 out of 210 fields had errors (1.43 pct.)

DATA FILE 1	DATA FILE 2
Record key field(s): (Rec. # 3) idcode = ML_J-2003-3300 res3 = 0	Record # 3 res3 = 9
Record key field(s): (Rec. # 11) idcode = ML_J-2003-3307 res3 = 0	Record # 11 res3 = 9
Record key field(s): (Rec. # 12) idcode = ML_J-2003-3308 age = 29	Record # 12 age = 39

After making correction in the “F” file, your data should be correct, or not? While your final data file should be correct, there is still a slim chance that it has errors. How is this possible? If by chance the same error was entered in both files (which can happen particularly if the same person enters the data in both files), you will not be able to identify the error. For uniformity, you should overwrite your existing file with the A_EX04F.REC file that is provided with the solution.

Exercise 5: Using an external file for Label blocks

At the end of this exercise you should be able to:

- a. Prepare a spreadsheet file to be saved as a CSV text formatted file acceptable for import into other database software, including an EpiData REC file
- b. Know how to prepare a QES and CHK file that accept an imported text file and can be used as an external label block
- c. Know how to access an external REC file from a CHK file for use as a label block

We have used label blocks to display Field value – Value label pairs. In our example, there were never more than ten values. We could have done with a few more, perhaps up to 20, perhaps even a few more, but there will certainly be an upper limit for the CHK file, we couldn't (or surely wouldn't be trying to) write thousands of lines into it – the CHK file has an upper limit: neither the QES file nor the CHK file can exceed the size of 64 KB (close to 66,000 key strokes is a lot of leverage). But apart from the file size limitations, it is also **inefficient and error-prone to write very large label blocks** into the Check file.

A quite common situation is that we have a list of districts or other administrative units to pick from and such lists can be quite long. As a general rule, one should always, whenever possible, use official lists and not making them up by one self. The latter is last resort if there is really no official list available. Such lists come in various formats and we will have to adapt them to our requirements. The common standard across a multitude of database formats are text files (not word processing files, they are not text files). Text files are the bare-bone elements of anything done on a computer: any key stroke on a computer can be expressed as a sequence of 8 bits (1 byte). In a text file, each keystroke (such as a letter) will take 1 byte, at least in the conventional standard (things are changing). Text files (traditionally in ASCII [*American Standard Code for Information Interchange*], now often extended to UTF-8 [*Unicode Transformation Format-8*]) will be accepted for import by virtually any decent software and can also be produced (exported to) by virtually any decent software. What is furthermore needed is a *delimiter*. A delimiter separates different things, two words for instance should be separated by a delimiter. In a standard text as this one here, the delimiter between two words is a space. In a data set, the space is not a good choice for a delimiter because there could be text fields that themselves have several words in one field, and this would then promptly lead to ambiguity and indeed errors. Tab stops are also not good delimiters because text editors often convert them into a series of spaces. That leaves the comma and the semicolon. Personally, we give preference to the semicolon because of the rarity of its use in text strings and its clear visibility. However, most popularity has seemingly been gained by Comma-Separated Values (*.CSV) files, and this is what we are going to use here. Depending on the language setting of your computer, a *.CSV file may actually has semi-colons as a delimiter, not commas as the name suggests. The delimiter is a comma on English keyboard settings (both American and British), but it is a semi-colon on

several continental European keyboards, depending how they write the decimal “point” which is a “point” in Anglophone countries, but often a comma in continental Europe. Thus to prevent ambiguity, *.CSV is not consistently “comma-separated”.

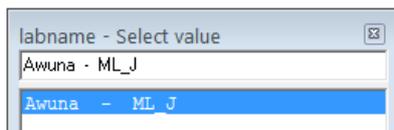
Example: the list with names and code comes as a spreadsheet

We include a spreadsheet formatted to Excel 2003 standard (a_ex05_namecode.xls):

	A	B
1	ML_J	Awuna
2	MS_D	Beitbridge
3	MC_A	Bindura
4	MN_G	Binga
5	ML_M	Birchenough
6	ML_I	Bonda
7	MS_G	Brunapeg
8	MW_J	Chegutu

Notice here that column A contains the laboratory code (LABCODE) and column B contains the laboratory name (LABNAME).

The first thing we have to think about is how our label should look in the end. If we remember our current label block as we had it in Exercise 4:

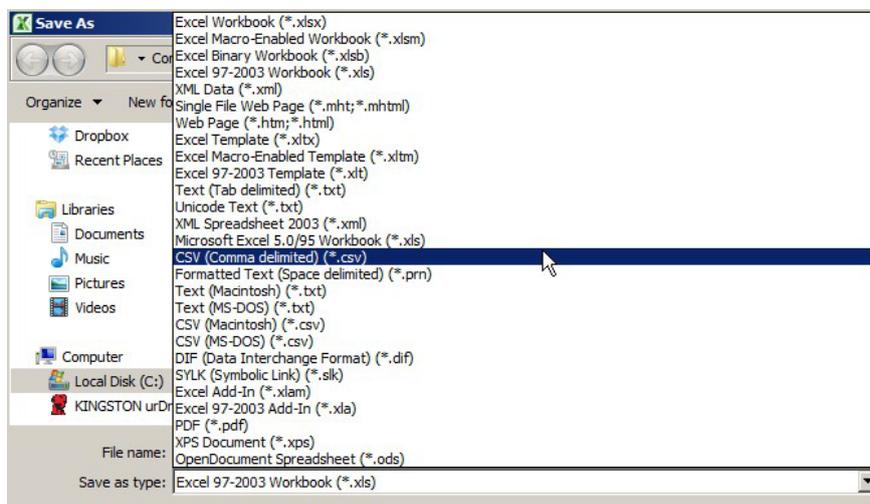
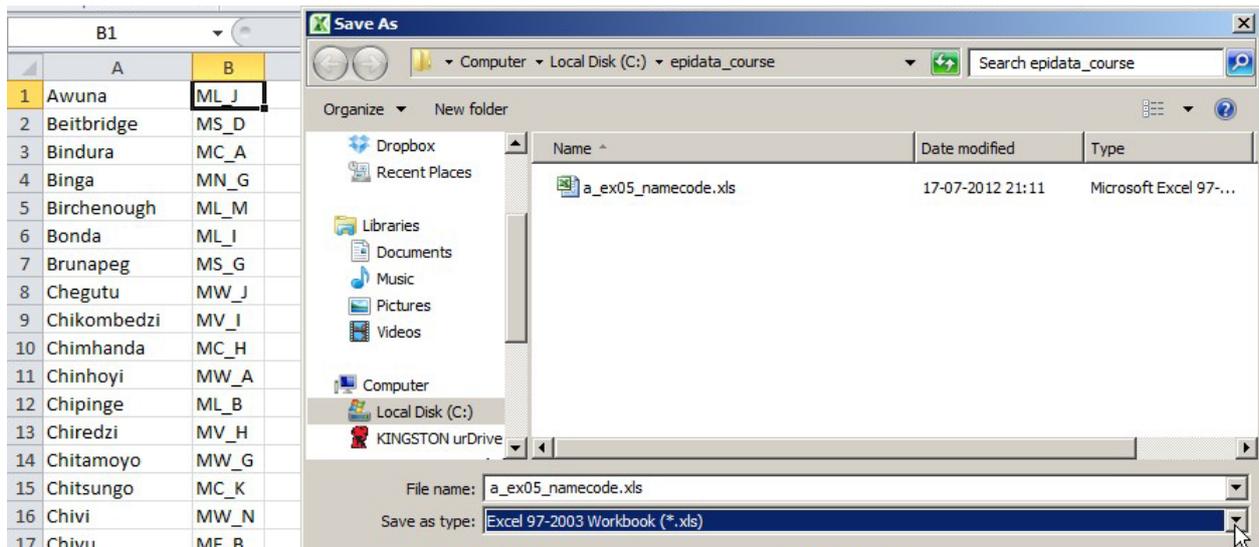


then we see that it was actually the other way around than the way we have it in the spreadsheet: the laboratory name is on the left and the laboratory code is on the right. We need thus to rearrange the columns in the spreadsheet accordingly to become:

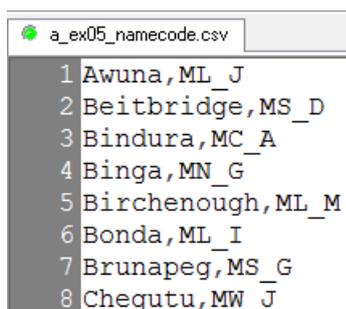
	A	B
1	Awuna	ML_J
2	Beitbridge	MS_D
3	Bindura	MC_A
4	Binga	MN_G
5	Birchenough	ML_M
6	Bonda	ML_I
7	Brunapeg	MS_G
8	Chegutu	MW_J

Then we “Save as” (depending on the spreadsheet software, Excel™ or LibreOffice Calc®) you may need to “Export”) “CSV Comma delimited (*.csv)” file.

Save the file as “a_ex05_namecode.csv”.



By all means, do not trust your spreadsheet to do it right: it is mandatory to check in your text editor software that you really got what you wanted:



Creating a "a_ex05_namecode.qes" file to provide the structure into which to import the text file

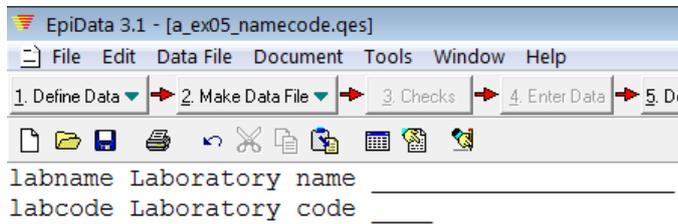
You have learned in the previous exercises that QES files provide the structure for EpiData files. We thus need a QES file into which the text file data can be read. This QES file will have only two fields:

```

labname
labcode

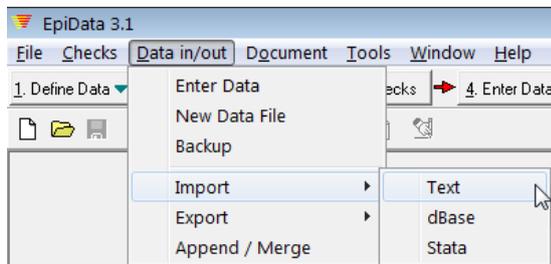
```

and in this sequence as everything has to be in this sequence: Field value then Value label. We have already previously defined that a field length of 20 for LABNAME suffices, while the length for LABCODE is 4, thus we make “a_ex05_namecode.qes” (same name, different extension):

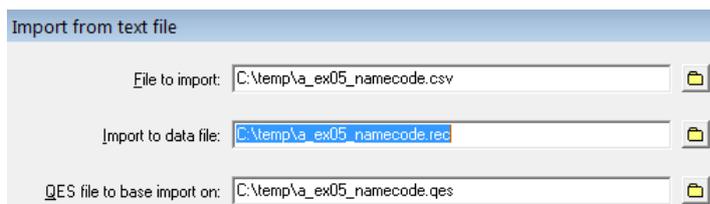


We don't need to make an “a_ex05_namecode.rec” file because the importing of the “a_ex05_namecode.csv” file into the “a_ex05_namecode.qes” file will actually create the “a_ex05_namecode.rec” file.

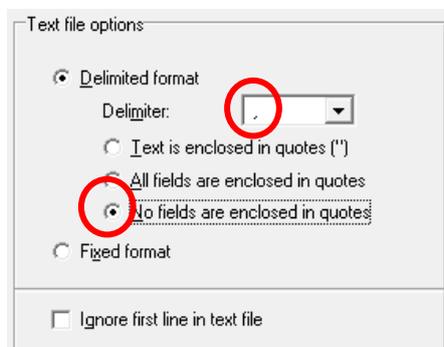
We go straight to:



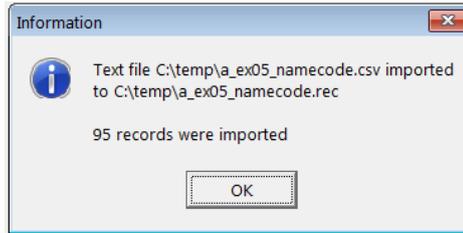
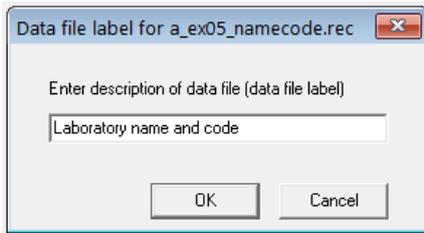
“Data in/out” | “Import” | “Text”. If we now pick the “a_ex05_namecode.csv” file, we see why it paid off to use the same file name for the QES file, with the correct naming all falls into place:



We then carefully check that all relevant boxes are selected / ticked / unticked:

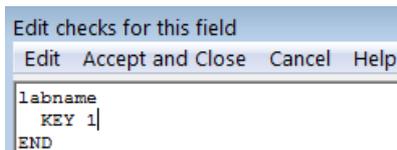


and then we get the *.REC file:



Creating a “a_ex05_namecode.chk” file

We need a *.CHK file. In order to enable the primary file to properly identify the fields in this file as Field value and Value label as the elements for the label block, LABNAME must become KEY 1 and LABCODE must become KEY 2. Add these Checks accordingly:

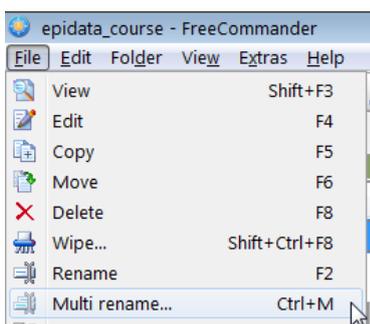


Updating the A_EX04.* to A_EX05.* files

As mentioned earlier there is a bug in EpiData during the export of CHK file. So, if we export an EpiData REC file to another EpiData REC file together with its CHK file, several errors appear in the export of CHK file.

EpiData Entry’s option to export data to EpiData format is excellent for a simple (unrelated) EpiData twin set of REC and CHK. It is too buggy for use with a relational database. We therefore do not recommend export.

Alternatively, we create copies of the existing triplet files and rename them appropriately in our file management software (Explorer or FreeCommander) and then do the required edits in all three files. You could use for instance the Multi rename tool in FreeCommander or an analogue in Explorer:



In summary, to make a label block out of an external file, the first pre-requisite is that the external file should be in the REC file format with two fields sequentially arranged to fit into prospective field values and value labels. Also, the two fields should be made key fields - KEY 1 and KEY 2 respectively. Hence, whatever be the format of our original database (Excel, Access, Text), we have to convert that into a REC file. One of the methods of doing it has been explained in this chapter which makes use of the limited options within EpiData

entry. However, there are much simpler/faster options of importing data and saving it as a REC file using EpiData Analysis which is discussed later in EpiData Analysis.

Tasks:

- o Finalize all to make an update A_EX05.QES / REC / CHK triplet*
- o Ensure that the file A_EX05_NAMECODE.REC serves properly as the new label block*

Solution to Exercise 5: Using an external file for Label blocks

Key Point(s):

- An external file for Labelblocks is more efficient when the list of values is very large and would result in a very long check file.

Tasks:

- o *Finalize all to make an update A_EX05.QES / REC / CHK triplet*
- o *Ensure that the file A_EX05_NAMECODE.REC serves properly as the new label block*

Solution

The A_EX05_NAMECODE.QES file:

```
labname Laboratory name _____
labcode Laboratory code ____
```

The A_EX05_NAMECODE.CHK file:

```
labname
  KEY 1
END
```

```
labcode
  KEY 2
END
```

The A_EX05.QES file:

This is the questionnaire for the laboratory register

```
labcode      Code of the laboratory 
idcode       Unique identifier 

labname      Name of the laboratory 
serno       Laboratory serial number  Enter 9001, 9002, ... if serial number is not unique and write data entry note (F5)
regdate     Registration date  Enter 01/01/1800 if missing
sex         Examinee's sex 
age         Examinee's age in years  Enter 999 if missing
reason      Examination reason 
res1        Result of specimen 1 
res1sc     Result of specimen 1 scanty 
res2        Result of specimen 2 
res2sc     Result of specimen 2 scanty 
res3        Result of specimen 3 
res3sc     Result of specimen 3 scanty 
```

The A_EX05.CHK file:

```
LABELBLOCK
  LABEL label_sex
    1 Female
    2 Male
    9 "Sex not recorded"
  END
  LABEL label_reason
    0 Diagnosis
    1 "Follow-up at 1 month"
    2 "Follow-up at 2 months"
    3 "Follow-up at 3 months"
    4 "Follow-up at 4 months"
    5 "Follow-up at 5 months"
    6 "Follow-up at 6 months"
    7 "Follow-up at 7 months or later"
    8 "Follow-up, month not stated"
    9 "Reason not recorded"
  END
  LABEL label_result
    0 Negative
    1 "1+ positive"
    2 "2+ positive"
    3 "3+ positive"
    4 "Positive, not quantified"
    5 "Scanty, not quantified"
    6 "Scanty, quantified"
    9 "Result not recorded"
  END
  LABEL label_scanty
    0 "Not applicable"
    1 "1 AFB per 100 OIF"
    2 "2 AFB per 100 OIF"
    3 "3 AFB per 100 OIF"
    4 "4 AFB per 100 OIF"
    5 "5 AFB per 100 OIF"
    6 "6 AFB per 100 OIF"
    7 "7 AFB per 100 OIF"
    8 "8 AFB per 100 OIF"
    9 "9 AFB per 100 OIF"
  END
END

BEFORE FILE
  DEFINE sernoTemp ____ GLOBAL
END

AFTER RECORD
  IF idcode=. THEN
    HELP "You cannot save a record without an identifier\n Please enter
all required information" TYPE=WARNING
    GOTO labname
  ENDIF
END

labcode
  NOENTER
END

idcode
```

```

KEY UNIQUE
NOENTER
END

labname
COMMENT LEGAL a_ex05_namecode.rec SHOW
MUSTENTER
TYPE COMMENT labcode
END

serno
MUSTENTER
AFTER ENTRY
    sernoTemp=serno
    IF serno<1000 THEN
        sernoTemp="0"+serno
    ENDIF
    IF serno<100 THEN
        sernoTemp="00"+serno
    ENDIF
    IF serno<10 THEN
        sernoTemp="000"+serno
    ENDIF
END
END

regdate
RANGE 01/01/2000 31/12/2005
LEGAL
    01/01/1800
END
MUSTENTER
AFTER ENTRY
    idcode=labcode+"-"+year(regdate)+"-"+sernoTemp
END
END

sex
COMMENT LEGAL USE label_sex SHOW
MUSTENTER
TYPE COMMENT
END

age
RANGE 0 125
LEGAL
    999
END
MUSTENTER
END

reason
COMMENT LEGAL USE label_reason SHOW
MUSTENTER
TYPE COMMENT
END

res1
COMMENT LEGAL USE label_result SHOW
MUSTENTER
TYPE COMMENT

```

```

AFTER ENTRY
  IF res1<>6 THEN
    res1sc=0
    GOTO res2
  ENDIF
END
END

res1sc
COMMENT LEGAL USE label_scanty SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
  IF (res1=6) AND (res1sc=0) THEN
    HELP "Values of res1 and res1sc not compatible. Please verify"
    GOTO res1
  ENDIF
END
END

res2
COMMENT LEGAL USE label_result SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
  IF res2<>6 THEN
    res2sc=0
    GOTO res3
  ENDIF
END
END

res2sc
COMMENT LEGAL USE label_scanty SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
  IF (res2=6) AND (res2sc=0) THEN
    HELP "Values of res2 and res2sc not compatible. Please verify"
    GOTO res2
  ENDIF
END
END

res3
COMMENT LEGAL USE label_result SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY
  IF res3<>6 THEN
    res3sc=0
    GOTO WRITE
  ENDIF
END
END

res3sc
COMMENT LEGAL USE label_scanty SHOW
MUSTENTER
TYPE COMMENT
AFTER ENTRY

```

```
IF (res3=6) AND (res3sc=0) THEN
  HELP "Values of res3 and res3sc not compatible. Please verify"
  GOTO res3
ENDIF
END
END
```

Exercise 6: Dealing with incomplete dates

At the end of this exercise you should be able to:

- a. Approximate dates when day and/or month is missing
- b. Create a date from three component fields (day, month, year)
- c. Make use of temporary variables in the check file to make calculations.

Date information is commonly collected, most commonly perhaps for the calculation of intervals or another purpose. However, information on the date is often incomplete. For instance, if a date of symptom onset is asked from a patient, the patient may remember only the year or only the year and the month. The date of onset is thus unknown, and an interval between the current doctor's visit (which might be known exactly) and date of symptom onset cannot be calculated and must remain unknown. Nevertheless, if for instance year and month of symptom onset are known (but not the day) and the date of the actual visit is known exactly, something is known, and an approximate interval could be provided. This exercise will offer a way to make such approximations.

As we should not leave it up to the data entry person to enter incorrect, approximated dates, with possibly arbitrary and changing rules, we must avoid requiring to enter a date, but rather split it into its components year, month, and day, and then reassemble it.

Let us assume that we wish to calculate the variables:

```
regexct      Exact registration date <dd/mm/yyyy>
regappr      Approximate registration date <dd/mm/yyyy>
```

from the fields:

```
regdd        Day of registration ##
regmm        Month of registration ##
regyy        Year of registration #####
```

The field REGEXCT will be an existing date within the legal range currently defined if all three components of the date are known else its value will be set to 01/01/1800. The field REGAPPR will be equal to REGEXCT if the latter is within the legal range. If the day only is unknown, the day value for REGAPPR will be 15 (mid-month), if both month and day are unknown, the month will be 07 and the day 01 (mid-year). If all three components are unknown, the value of REGAPPR will be set to 01/01/1800.

We will impose a hierarchical structure, i.e., once the month is unknown, it does not matter even if you know the day – even if entered as such – it does not make sense, and if the year is unknown, neither a known month nor known day will affect the calculation.

Thus, we will replace the current field REGDATE with the above five fields, three of which (REGDD, REGMM, and REGYY) are entered (MUSTENTER fields) while the other two (REGEXCT and REGAPPR) are calculated from the former three (and will thus be NOENTER Fields).

In addition, we will add another variable that measures the quality of the calculated date information, and that we might name:

```
regqual      Quality of registration date #
```

This field can take the values 0, 1, 2, and 3, where 0 (zero) indicates that none is known, and 3 that all three date components are known.

For this task, we use temporary variables that we have briefly introduced in an earlier exercise. Although for this specific example, it would be possible to solve it without such variables, they make things easier and more transparent. Temporary variables can have a field length of 16 and they do not appear in the QES file and will not become part of the dataset (see our earlier exercise). They will be used in the Check file for internal calculations in a BEFORE FILE block as in:

```
BEFORE FILE
  DEFINE regddTemp ##
  DEFINE regmmTemp ##
  DEFINE regyyTemp ####
END
```

BEFORE FILE means exactly what it says: Before anything is entered into the file, these variables are created: we have used this in an earlier exercise. There is also a CHK file command BEFORE RECORD (look up in the Help file what these commands do).

To create a date from the three component fields, the CHK file command is:

```
fulldate=DATE(varday,varmonth,varyear)
```

Tasks:

- o Create an A_EX06.* triplet (using the A_EX05.* files as the starting point). The questionnaire should display all the calculated variables.*
- o Try to preserve the data you have (in A_EX05.REC), accepting that you lose the information in the field REGDATE, which is easier to update than recreating the entire file.*
- o Edit the A_EX06.CHK file to make the calculations. Note that you will need to define temporary variables for this task.*
- o Update the data file to check the functionality*

Solution to Exercise 6: Dealing with incomplete dates

Key Point(s):

- Approximating dates in a systematic way will enable you to approximate intervals e.g. between date of symptom onset and visit to the health facility.
- Do not leave it to the data entry person to enter incorrect or approximate dates.

Tasks:

- o *Create an A_EX06.* triplet (using the A_EX05.* files as the starting point). The questionnaire should display all the calculated variables.*
- o *Try to preserve the data you have (in A_EX05.REC), accepting that you lose the information in the field REGDATE, which is easier to update than recreating the entire file.*
- o *Edit the A_EX06.CHK file to make the calculations. Note that you will need to define temporary variables for this task.*
- o *Update the data file to check the functionality*

Solution

The A_EX06.QES file:

This is the questionnaire for the laboratory register

labcode	Code of the laboratory	<input type="text"/>	
idcode	Unique identifier	<input type="text"/>	
regexact	Exact registration date	<input type="text"/>	Set to 01/01/1800 if any missing
regappr	Approximate registration date	<input type="text"/>	Set to 01/01/1800 if year missing
regqual	Quality of registration date	<input type="text"/>	
labname	Name of the laboratory	<input type="text"/>	
serno	Laboratory serial number	<input type="text"/>	Enter 9001, 9002, ... if serial number is not unique and write data entry note (F5)
regdd	Registration day	<input type="text"/>	Enter 99 if missing
regmm	Registration month	<input type="text"/>	Enter 99 if missing
regyy	Registration year	<input type="text"/>	Enter 9999 if missing
sex	Examinee's sex	<input type="text"/>	
age	Examinee's age in years	<input type="text"/>	Enter 999 if missing
reason	Examination reason	<input type="text"/>	
res1	Result of specimen 1	<input type="text"/>	
res1sc	Result of specimen 1 scanty	<input type="text"/>	
res2	Result of specimen 2	<input type="text"/>	
res2sc	Result of specimen 2 scanty	<input type="text"/>	
res3	Result of specimen 3	<input type="text"/>	
res3sc	Result of specimen 3 scanty	<input type="text"/>	

The A_EX06.CHK file (pertinent parts only):

```

LABELBLOCK
  LABEL label_sex
    1 Female
    2 Male
    9 "Sex not recorded"
  END
  LABEL label_reason
    0 Diagnosis
    1 "Follow-up at 1 month"
    2 "Follow-up at 2 months"
    3 "Follow-up at 3 months"
    4 "Follow-up at 4 months"
    5 "Follow-up at 5 months"
    6 "Follow-up at 6 months"
    7 "Follow-up at 7 months or later"
    8 "Follow-up, month not stated"
    9 "Reason not recorded"
  END
  LABEL label_result
    0 Negative
    1 "1+ positive"
    2 "2+ positive"
    3 "3+ positive"
    4 "Positive, not quantified"
    5 "Scanty, not quantified"
    6 "Scanty, quantified"
    9 "Result not recorded"
  END
  LABEL label_scanty
    0 "Not applicable"
    1 "1 AFB per 100 OIF"
    2 "2 AFB per 100 OIF"
    3 "3 AFB per 100 OIF"
    4 "4 AFB per 100 OIF"
    5 "5 AFB per 100 OIF"
    6 "6 AFB per 100 OIF"
    7 "7 AFB per 100 OIF"
    8 "8 AFB per 100 OIF"
    9 "9 AFB per 100 OIF"
  END
END

BEFORE FILE
  DEFINE sernoTemp _____ GLOBAL
  DEFINE regddTemp ##
  DEFINE regmmTemp ##
  DEFINE regyyTemp ####
END

AFTER RECORD
  IF idcode=. THEN
    HELP "You cannot save a record without an identifier\n Please enter all
required information" TYPE=WARNING
    GOTO labname
  ENDIF
END

labcode
  NOENTER
END

idcode
  KEY 1
  NOENTER
END

regexct
  NOENTER

```

```

END

regappr
  NOENTER
END

regqual
  NOENTER
END

labname
  COMMENT LEGAL a_ex05_namecode.rec SHOW
  MUSTENTER
  TYPE COMMENT labcode
END

serno
  MUSTENTER
  AFTER ENTRY
    sernoTemp=serno
    IF serno<1000 THEN
      sernoTemp="0"+serno
    ENDIF
    IF serno<100 THEN
      sernoTemp="00"+serno
    ENDIF
    IF serno<10 THEN
      sernoTemp="000"+serno
    ENDIF
  END
END

regdd
  RANGE 1 31
  LEGAL
    99
  END
  MUSTENTER
END

regmm
  RANGE 1 12
  LEGAL
    99
  END
  MUSTENTER
  REPEAT
END

regyy
  RANGE 2000 2005
  LEGAL
    9999
  END
  MUSTENTER
  REPEAT
  AFTER ENTRY
    regddTemp=regdd
    regmmTemp=regmm
    regyyTemp=regyy
    IF (regdd=99) or (regmm=99) or (regyy=9999) THEN
      regexct="01/01/1800"
    ELSE
      regexct=date(regddTemp,regmmTemp,regyyTemp)
      regappr=regexct
      regqual=3
    ENDIF
    IF regdd=99 THEN

```

```

    regddtemp=15
    regqual=2
ENDIF
IF regmm=99 THEN
    regddTemp=01
    regmmTemp=07
    regqual=1
ENDIF
IF regyy=9999 THEN
    regddTemp=01
    regmmTemp=01
    regyyTemp=1800
    regqual=0
ENDIF
regappr=date(regddTemp,regmmTemp,regyyTemp)
idcode=labcode+"-"+regyyTemp+"-"+sernoTemp
END
END

```

...

A completed A_EX06.REC record:

This is the questionnaire for the laboratory register

labcode	Code of the laboratory	<input type="text" value="MV_I"/>	
idcode	Unique identifier	<input type="text" value="MV_I-2003-0324"/>	
regexct	Exact registration date	<input type="text" value="01/01/1800"/>	Set to 01/01/1800 if any missing
regappr	Approximate registration date	<input type="text" value="15/10/2003"/>	Set to 01/01/1800 if year missing
regqual	Quality of registration date	<input type="text" value="2"/>	

labname	Name of the laboratory	<input type="text" value="Chikombedzi"/>	
serno	Laboratory serial number	<input type="text" value="324"/>	Enter 9001, 9002, ... if serial number is not
regdd	Registration day	<input type="text" value="99"/>	Enter 99 if missing
regmm	Registration month	<input type="text" value="10"/>	Enter 99 if missing
regyy	Registration year	<input type="text" value="2003"/>	Enter 9999 if missing
sex	Examinee's sex	<input type="text" value="1"/>	Female
age	Examinee's age in years	<input type="text" value="23"/>	Enter 999 if missing
reason	Examination reason	<input type="text" value="0"/>	Diagnosis
res1	Result of specimen 1	<input type="text" value="0"/>	Negative
res1sc	Result of specimen 1 scanty	<input type="text" value="0"/>	
res2	Result of specimen 2	<input type="text" value="1"/>	1+ positive
res2sc	Result of specimen 2 scanty	<input type="text" value="0"/>	
res3	Result of specimen 3	<input type="text" value="6"/>	Scanty, quantified
res3sc	Result of specimen 3 scanty	<input type="text" value="3"/>	3 AFB per 100 OIF

Exercise 7: Keeping track of data entry time

At the end of this exercise you should be able to:

- a. Edit the check file, and use temporary variables to calculate the amount of time required for data entry.

In this exercise you will learn to calculate the number of seconds which are required to complete data entry for one record. A field to retain this value as part of the database thus needs to be added to the questionnaire.

If you prepare a study, you should make it a rule to enter several hundred records by yourself (with the help of a colleague). This is important for two reasons:

- o Identify weaknesses in the data entry form and the CHK file
- o Recording the time needed to enter the information

You need the information on recording time to know what you can expect from another data entry person. This is critical for making your budget. It is obviously not satisfactory to pay a data entry person according to the time the person claims to need: some people are slow and they should not receive the same remuneration as the faster person. If you take your achievement as the basis, you can objectively pay for the time required if a data entry person works at your speed: those slower than you will lose, those faster will win, and that is how it should be.

We can use EpiData Entry to monitor our data entry time and make a permanently available record that we can later analyze. In the `EpiData\samples\` folder you find three files:

```
DateTime.qes  
DateTime.rec  
DateTime.chk
```

These three files are the basis to adjust our own sample files together with the **About time** in the EpiData Entry Help file.

There are two important things you need to know about dates.

- EpiData works internally with dates as date numbers, counting the number of days since 31st December 1899, which has the day number 1. This is referred to as the anchor date. The date 15th October 2000 has, for example, the day number 36814. The advantage of day numbers is that it makes it easy to perform calculations with dates, e.g. adding 7 days to a date or counting the number of days from a specific date to today.
- EpiData understands dates in European date format only i.e, `dd/mm/yyyy`. So a date `01/07/2012` is understood as 1 July, 2012 and not 7 January, 2012. Date constants can be defined in two ways: either by `"15/03/1977"` or by `date(15,3,1977)` which both will produce the date 15th March 1977.

We are going to make use of computer's clock. EpiData Entry has a function `NOW` which records the exact time of the computer. This time consists of the date and the time of the day. `NOW` writes the information on the date into the integer part of a field and the time of the day

as a fraction of the 24-hour clock into the fractional part of the field. If we define thus such a field for the file and start counting the time when a new record is opened:

```
BEFORE FILE
  DEFINE StartTime #####.#####
END

BEFORE RECORD
  StartTime=NOW
END
```

We could of course write the field `StartTime` into the questionnaire as a `NOENTER` field and then we would get, as an example:

39650.864672

`NOW` gives the computer time right at this point, writing the date as the integer part and the fraction of the day as the fractional part. The computer stores the date as the number of days passed since the internal anchor date, 31 December 1899. If we divide the above integer part of the number by the average number of days per year we get $39650/365.25=108.56$, that is a bit more than 108.5 years after the anchor date, or some time in July 2008. What about the fractional part? If we multiply the 24-hour day by this fraction we get $0.864672*24=20.75$ which is roughly a quarter to nine in the evening, but it is much more precise than to the quarter of the hour (6 digits!) because what we really want is to have it expressed in seconds and one day has 86,400 seconds. To get the rounding properly we use even 6 rather than the bare minimum of 5 digits.

As the database only needs to retain the number of seconds required to complete one record, the number above is thus just something we need the `CHK` file to calculate in the background.

Assuming that we have a field `SECONDS` in the data file, we would then write after entering the value for the last field:

```
res3sc
  COMMENT LEGAL USE label_scanty SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
    IF seconds=. THEN
      seconds=(NOW-StartTime)*86400
    ENDIF
  END
END
```

This `NOW` is a different `NOW` from the beginning (computer clock!) and all we do is to revert the difference between the two time points into seconds.

If we do it as above, the clock will stop right after the value of the last field `RES3SC` has been entered and that value (number of seconds) will be written into the field `SECONDS`. Even when returning to the record, the original value in the field `SECONDS` will not change. If we were to take out the:

```
IF seconds=. THEN
  seconds=(NOW-StartTime)*86400
ENDIF
```

then the number of seconds would change to a new value if going again later through the record, and the value then might be lower. Ideally, however, the number of SECONDS would need to be cumulative, i.e. if a record is re-visited, the additional time during the revisit would be added to the pre-existing time. In other words, the time for corrections after validation would be added. The change required to accomplish the latter is to delete it from the AFTER ENTRY statement in the RES3SC field:

```
res3sc
  COMMENT LEGAL USE label_scanty SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
  IF seconds=. THEN
    seconds=(NOW-StartTime)*86400
  ENDIF
  END
END
```

Instead, the statements are integrated into the AFTER RECORD statement and extended to provide accumulation of time. *Note that this is a better choice than the previous option as the amount of time required to save the record would also be included in the calculations now.*

```
AFTER RECORD
  IF idcode=. THEN
    HELP "You cannot save a record without an identifier\n Please enter
all required information" TYPE=WARNING
    GOTO labname
  ENDIF
  IF seconds=. THEN
    seconds=(NOW-StartTime)*86400
  ELSE
    seconds=seconds+(NOW-StartTime)*86400
  ENDIF
END
```

Let us summarize the rationale of this approach once again.

Before the entry of a record is started, StartTime gets a value corresponding to the time of the clock at that point in time in your computer using the function NOW. It will be in the format 39814.25678 – the integer part refers to the date (converted into a number after subtracting the anchor date from current date) and the decimal part refers to the fraction of time in the day. Similarly, after the entry is complete, we will record the time at that point, again in the same format 39814.26677. The difference between the two gives the time in terms of fraction of the day = (39814.26677-39814.25678)=0.00999; this fraction, when multiplied by 24 (number of hours in a day) will give time duration in hours; when multiplied by 1440 (number of minutes in a day) will give time duration in minutes. Since we need the duration in seconds, we have multiplied the fraction by 86400 (the number of seconds in a day).

However, an even more informative approach is to have both the information about time required to enter one record for the first time and the information about the total amount of time (cumulatively) spent on a records, i.e. including the time spent on corrections. In the analysis one can then thus determine how much time is spent on entering one record, and how

much additional time on correcting the record if a discordance must be resolved after validation (by subtracting the value in the field for first entry from that of the cumulative time). This is important because it has been shown that working faster is accompanied by making more errors, thus requiring revisiting more records again.

Note on formatting: The choice of a mixture of lower-case and upper-case letters (eg, in `StartTime`) is for easier visual recognition only. As mentioned earlier, EpiData Entry is not case-sensitive. This makes it particularly powerful as you can make use of formatting to visual ease. As a general rule, if you let EpiData Entry make the choices for you, commands are capitalized, all the rest is not. As for field names, lower-case is always preferred. While it does not matter in formatting, you can force its format in the REC file to any option you prefer, but we give preference to lower case (go to “File” “Options” “Create data file”).

Task:

- o Start with the A_EX06 QES-REC-CHK files, save them as A_EX07 QES-REC-CHK and revise them accordingly. Don't just retype the above, try to consider the logic of it!***
- o Make two NOENTER fields, one that calculates the data entry time for the first entry (that will not change by revisiting the records) and another field that calculates the cumulative time resulting from one or more re-visits of the record.***
- o Enter some data to check the functionality.***

Solution to Exercise 7: Keeping track of data entry time

Key Point(s)

- The amount of time required to enter a record is critical for making a budget, and payment can be objectively made.

Task:

- o *Start with the A_EX06 QES-REC-CHK files, save them as A_EX07 QES-REC-CHK and revise them accordingly. Don't just retype the above, try to consider the logic of it!*
- o *Make two NOENTER fields, one that calculates the data entry time for the first entry (that will not change by revisiting the records) and another field that calculates the cumulative time resulting from one or more re-visits of the record.*
- o *Enter some data to check the functionality.*

Solution

The A_EX07.QES file:

This is the questionnaire for the laboratory register

labcode	Code of the laboratory	<input type="text"/>	
idcode	Unique identifier	<input type="text"/>	
regexct	Exact registration date	<input type="text"/>	Set to 01/01/1800 if any missing
regappr	Approximate registration date	<input type="text"/>	Set to 01/01/1800 if year missing
regqual	Quality of registration date	<input type="text"/>	
seconds	Number of seconds for record	<input type="text"/>	
cumsecs	Cumulative number of seconds	<input type="text"/>	

labname	Name of the laboratory	<input type="text"/>	
serno	Laboratory serial number	<input type="text"/>	Enter 9001, 9002, ... if serial number is r
regdd	Registration day	<input type="text"/>	Enter 99 if missing
regmm	Registration month	<input type="text"/>	Enter 99 if missing
regyy	Registration year	<input type="text"/>	Enter 9999 if missing
sex	Examinee's sex	<input type="text"/>	
age	Examinee's age in years	<input type="text"/>	Enter 999 if missing
reason	Examination reason	<input type="text"/>	
res1	Result of specimen 1	<input type="text"/>	
res1sc	Result of specimen 1 scanty	<input type="text"/>	
res2	Result of specimen 2	<input type="text"/>	
res2sc	Result of specimen 2 scanty	<input type="text"/>	
res3	Result of specimen 3	<input type="text"/>	
res3sc	Result of specimen 3 scanty	<input type="text"/>	

The A_EX07.CHK file (only the pertinent parts):

```

...
BEFORE FILE
  DEFINE sernoTemp _____ GLOBAL
  DEFINE regddTemp ##
  DEFINE regmmTemp ##
  DEFINE regyyTemp ####
  DEFINE StartTime #####.#####
END

BEFORE RECORD
  StartTime=NOW
END

AFTER RECORD
  IF idcode=. THEN
    HELP "You cannot save a record without an identifier\n Please enter all
required information" TYPE=WARNING
    GOTO labname
  ENDIF
  IF seconds=. THEN
    seconds=(NOW-StartTime)*86400
  ENDIF
  IF cumsecs=. THEN
    cumsecs=(NOW-StartTime)*86400
  ELSE
    cumsecs=cumsecs+(NOW-StartTime)*86400
  ENDIF
END

```

A completed record

This is the questionnaire for the laboratory register

labcode	Code of the laboratory	<input type="text" value="ML_J"/>	
idcode	Unique identifier	<input type="text" value="ML_J-2003-3298"/>	
regext	Exact registration date	<input type="text" value="26/10/2003"/>	Set to 01/01/1800 if any missing
regappr	Approximate registration date	<input type="text" value="26/10/2003"/>	Set to 01/01/1800 if year missing
regqual	Quality of registration date	<input type="text" value="3"/>	
seconds	Number of seconds for record	<input type="text" value="8"/>	
cumsecs	Cumulative number of seconds	<input type="text" value="25"/>	

labname	Name of the laboratory	<input type="text" value="Awuna"/>	
serno	Laboratory serial number	<input type="text" value="3298"/>	Enter 9001, 9002, ... if serial number is no
regdd	Registration day	<input type="text" value="26"/>	Enter 99 if missing
regmm	Registration month	<input type="text" value="10"/>	Enter 99 if missing
regyy	Registration year	<input type="text" value="2003"/>	Enter 9999 if missing
sex	Examinee's sex	<input type="text" value="1"/>	
age	Examinee's age in years	<input type="text" value="35"/>	Enter 999 if missing
reason	Examination reason	<input type="text" value="5"/>	
res1	Result of specimen 1	<input type="text" value="0"/>	
res1sc	Result of specimen 1 scanty	<input type="text" value="0"/>	
res2	Result of specimen 2	<input type="text" value="0"/>	
res2sc	Result of specimen 2 scanty	<input type="text" value="0"/>	
res3	Result of specimen 3	<input type="text" value="9"/>	
res3sc	Result of specimen 3 scanty	<input type="text" value="0"/>	

Exercise 8: Data safety and security

At the end of this exercise you should be able to:

- a. Backup and encrypt a data file on an external medium.

Nothing should be more valuable to you than the safety of your data. Unfortunately, people often learn that they should have backed up their data only once they have made the painful experience of having lost them.

EpiData Entry can be very helpful and forcing you to back up your data when you close a file. Whether you back up your data to your hard drive (not a good idea in case of hard drive failure) or an external medium, an additional concern is data security. Let's assume you back up your data to a USB flash memory stick and subsequently you lose it. Anybody who finds your stick can look at your data. Perhaps your data contain confidential information that should not be readable by anybody who is not authorized.

EpiData Entry offers password-only accessible encryption that cannot be broken without knowing the password you used for encrypting them. We will make a simple EpiData Entry triplet to show how to back up your data with and without encryption to an external medium.

We will use the following simple dataset of the EpiData promoters:

Name	First Name	Country
Rieder	Hans L	Switzerland
Chiang	Chen-Yuan	Taiwan
Hinderaker	Sven Gudmund	Norway
Katamba	Achilles	Uganda
Kumar	Ajay	India
Tun	Zaw Myo	Myanmar
Nguyen	Binh Hoa	Vietnam
Zwahlen	Marcel	Switzerland

Our A_EX08.QES will look as follows:

Questionnaire for back-up data

```
name          <AAAAAAAAAAAAAAAA>
firstname     <AAAAAAAAAAAAAAAA>
country       <AAAAAAAAAAAAAAAA>
```

Make a REC file and a Check file. In the A_EX08.CHK file make all fields MUSTENTER fields, and then open the Check file which should look as follows:

```
name
  MUSTENTER
END
```

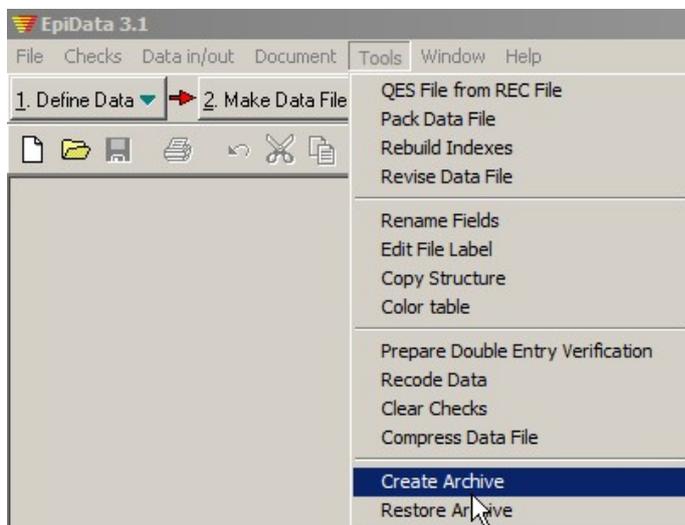
```
firstname
  MUSTENTER
END
```

```
country
  MUSTENTER
END
```

Backing up and encrypting your file

We will first demonstrate the menu option for doing this and then how it can be written down as a command in the CHK file.

Let us say, we want to back-up our course folder (C:\epidata_course). Go to Tools in the menu bar and click Create Archive



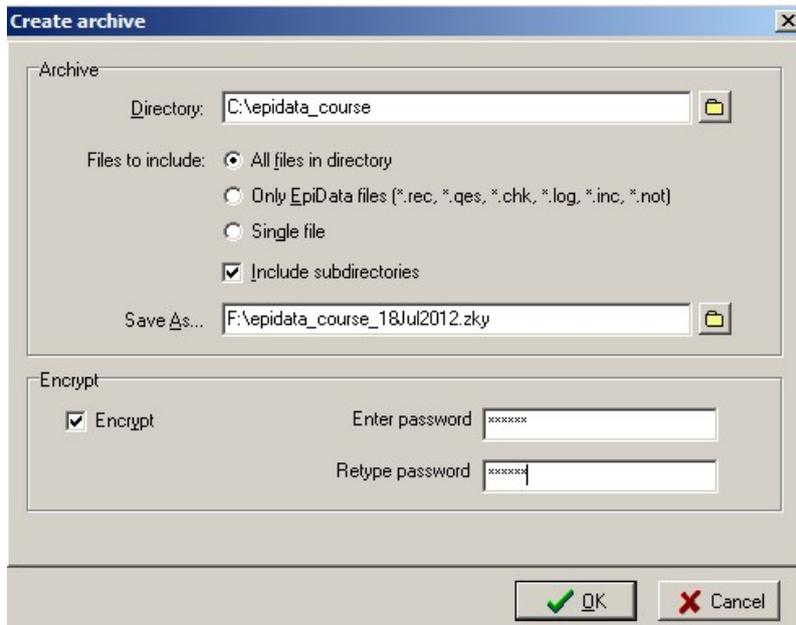
Browse and choose the appropriate directory to be archived.

Tick the options appropriately as preferred. We tick the options “All files in the directory” and “Include subdirectories”.

Browse and choose a location where you want to archive

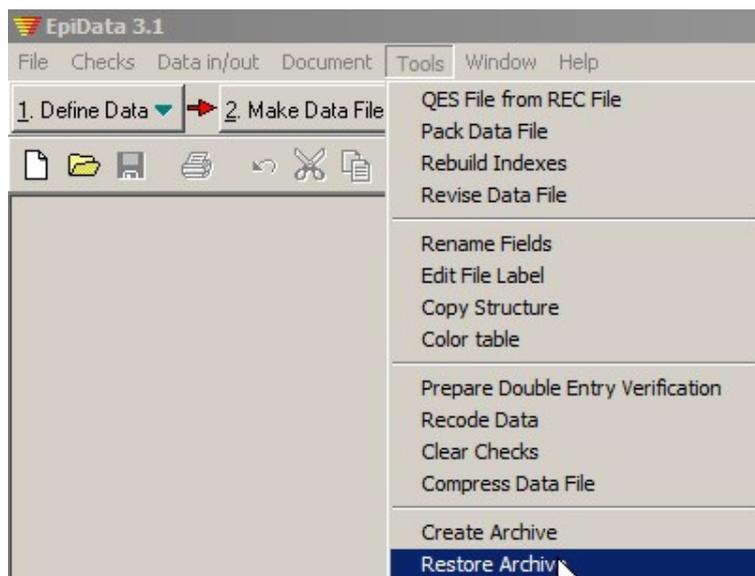
Tick Encrypt and provide a password.

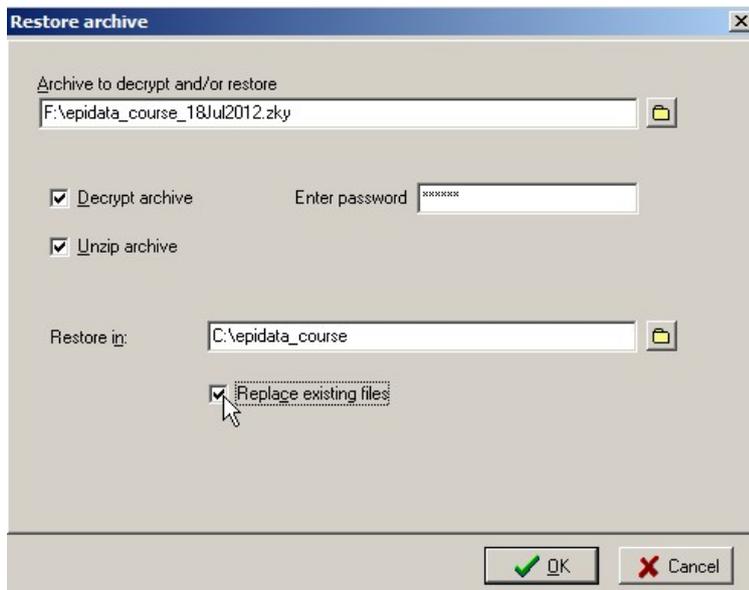
Note the naming which ends with date when this was archived and the extension .ZKY indicating that this is an encrypted file.



Re-Opening your back-up file

In “Tools” you find “Restore archive” and you are prompted to provide the path and name of your archived and encrypted file, to enter your password, and to provide the path where to restore it, with the option to overwrite any existing files. (Examine the following screen shots)





Let us now look at the CHK commands which can do the above steps. Assume that your external back-up device takes drive E: and has a folder named PROJECT on the drive. The command we add in the check file is then:

```
name
  MUSTENTER
END
```

```
firstname
  MUSTENTER
END
```

```
country
  MUSTENTER
END
```

```
AFTER FILE
  BACKUP e:\project ENCRYPT filename mypassword TODAY
END
```

Note that we have used two commands above.

The command BACKUP archives your EpiData triplet in the indicated folder. The E:\PROJECT is the folder on the drive of your memory stick which you must have created beforehand.

The command ENCRYPT helps to encrypt the data and pass-word protect the file. FILENAME is the name of the encrypted file and should best be what it is now, i.e. A_EX08 and “mypassword” is a personally chosen password.

The back-up file will have the name A_EX08.ZKY and will contain not only the REC file but the entire QES-REC-CHK triplet.

Note: *If you forget your password, there will be no way to ever open your backup file.* Of course, you can always see it in your original Check file if you choose to keep that. The TODAY function is optional but a good idea to ensure that the current computer date becomes part of the file name. It must be written after the password.

Note: The command BACKUP will back up all files in that folder. Thus, if you wish to backup up only the files A_EX06.* you must first copy those EpiData files that you wish to back up into another empty folder (such as the c:\temp folder after emptying it).

Task:

- o Create the QES-REC-CHK triplet, edit the Check file, enter the data above and restore the files.*

Solution to Exercise 8: Data safety

Key Point(s):

- You should always backup data files to avoid loss of important information when the hard drive fails.
- Data files with confidential information saved on an external medium should always be encrypted so that in case you lose it, nobody will be able to read the information.

Task:

- o *Create the QES-REC-CHK triplet, edit the Check file, enter the data above and restore the files.*

Solution:

This is the A_EX08.QES file:

Questionnaire for back-up data

```
name          <AAAAAAAAAAAAAAAA>
firstname     <AAAAAAAAAAAAAAAA>
country       <AAAAAAAAAAAAAAAA>
```

This is the A_EX08.CHK file:

```
name
  MUSTENTER
END
```

```
firstname
  MUSTENTER
END
```

```
country
  MUSTENTER
END
```

```
AFTER FILE
  BACKUP e:\project ENCRYPT a_ex08 mypassword TODAY
END
```

Of course, the password used here (“mypassword”) should be replaced by your own.