1529

Bell Laboratories

Cover Sheet for Technical Memorandum

The information contained herein is for the use of employees of Bell Laboratories and is not for publication (see GEI 13.9-3)

Title- Shell Commands for Relational Operations on Fielded Files (SCROFF)

Date- January 2, 1980

TM- 80-5424-1
5424-800102.01TM

Other Keywords- UNIX
Data Management

Author(s)	Location and Room	Extension
S. M. Salchenberger	IH 4G-331	4886

Charging Case- 49358-001

Filing Case- 40016-100

ABSTRACT

UNIX™ users having data management requirements must either be satisfied with existing UNIX tools such as awk, grep, and join, or must employ one of a number of integrated data base management systems. The former choice usually represents a compromise in terms of the power of the tools provided (since UNIX is not a data base system), while the latter often forces the user to relinquish ownership and control over his data.

This memorandum describes a set of shell level commands (called SCROFF) designed to bridge the gap between the aforementioned choices. SCROFF provides a more than relationally complete interface to ordinary tabular files without altering them in any way. The user therefore gains the advantages of powerful data base tools without losing control over his data.

THIS DOCUMENT CONTAINS NEITHER RECOMMENDATIONS NOR CONCLUSIONS OF THE FEDERAL COMMUNICATIONS COMMISSION. IT HAS BEEN DISTRIBUTED FOR INFORMATIONAL PURPOSES ONLY AND THE CONTENTS THEREOF MAY NOT BE DISCLOSED TO ANY THIRD PARTY WITHOUT THE WRITTEN PERMISSION OF THE FEDERAL COMMUNICATIONS COMMISSION.

Pages Text 7 Other 22 Total 29

No. Figures 0 No. Tables 0 No. Refs. 11

Y

BELL TELEPHONE LABORATORIES, INC.

TM-80-5424-1

DISTRIBUTION
(REFER GEI 13.9-3)

COMPLETE MEMORANDUM TO

CORRESPONDENCE FILES

OFFICIAL FILE COPY
PLUS ONE COPY FOR
EACH ADDITIONAL FILING
CASE REFERENCED

DATE FILE COPY
(FORM 2-1328)

10 REFERENCE COPIES

+AHORA,ADARSH K
+BAGLEY,RICHARD T
BOESE,J O
+CHANG,JO-MEI
CLEMENT,G F
+COOPER,MICHAEL H
+FREDERICK,DONALD H,JR
+FREEMAN,G E
+GERSIC,RAYMOND L
GILOTH,PAUL K
+HIGGINS,SARAH J
HUTTENHOFF,JOHN H
JAEGER,E J,JR
+JEFFREY,H JOEL
KIENZLE,H G
+KNUDSON,DALE O
+KRIST,JAMES T
+LARTZ,GEOLL W
MARTERSTECK,K E
+MEDHEKAR,RANJANA A
POOLE,N K
+ROBERTSON,ALAN L
+SCHUTTE,L J
<SMITH,M T,JR
+SMITH,ROBERT B
TOMKO,L A
WITSKEN,J R
YODER,KEITH W
28 NAMES

COVER SHEET ONLY TO

CORRESPONDENCE FILES

4 COPIES PLUS ONE
COPY FOR EACH FILING
CASE

ABATE,JOSEPH
AHO,ALFRED V
AHRENS,RAINER B
AHUJA,SUDHIR R
ALBERALLA,RICHARD J
ALCALAY,D
ALKONS,FREDERICK
ALLEN,R C
ALLEN,ROBERT B
AMITAY,N

COVER SHEET ONLY TO

AMOSS,JOHN J
AMRON,IRVING
ANDERSON,KATHRYN J
ANDERSON,MILTON M
ANDRAULT DE LANGERON,D
APPELBAUM,MATTHEW A
ATAL,BISHNU S
BACH,MAURICE J
BAGGA,YUDHVEER S
BAIN,WILLIAM LAMAR,JR
BALDWIN,GEORGE L
BALLARD,E D,JR
BARBATO,ROBERT R
BAROPSKY,ALLEN
BARON,ROBERT V
BARR,W J
BAUER,BARBARA T
BAUER,WOLFGANG F
BEACHY,MILTON
BECKER,RICHARD A
BEDNAR,JOSEPH A,JR
BENCO,DAVID S
BENISCH,JEAN
BENNETT,RAYMOND W
<BERGLAND,G D
BERNHARDT,RICHARD C
BERNSTEIN,DANIELLE B
BERNSTEIN,L
BERRIMAN,EDWARD R
BERZINS,ALEXANDER H
BEYER,JEAN-DAVID
BILLINGTON,MARJORIE J
BILOWOS,R M
BIREN,IRMA B
BISHOP,VERONICA L
BLAKE,GARY D
BLAZIER,S D
BLEICHER,E
BLEIER,JOSEF
BLINE,J C
BLOSSER,PATRICK A
BLUM,MARION
BODEN,F J
BOGART,THOMAS G
BOIVIE,RICHARD H
BONANNI,L E
BOSWELL,PAULA S
BOWYER,L RAY
BOYCE,W M
BOYER,PHYLLIS J
BOYLE,GERALD C
BRADLEY,M HELEN
BREEN,R S
BRIGGS,GLORIA A
BROAD,MARTHA M
BROSS,JEFFREY D
BROYMAN,INNA
BROWN,ELLINGTON L
BROWN,W R
BROWN,W STANLEY
BURG,F M
BURIC,MILORAD R

COVER SHEET ONLY TO

BUROFF,STEVEN J
BYRNE,EDWARD R
CANADAY,RUDD H
CAREY,J H
CARTER,DONALD H
CASPERS,BARBARA E
CATO,H L
CAVINESS,JOHN D
CERMAK,I A
CHAI,D T
CHAMBERS,B C
CHAMBERS,J M
CHANG,CHUNG W
CHAPMAN,W P,JR
CHEN,PING C
CHIANG,T C
CHI,M C
CHONG,PAEE
CHRIST,C W,JR
CLAYTON,D P
CLINE,LAUREL M I
COHEN,AARON S
COHEN,DAVID
COHEN,HARVEY
<COLE,LOUIS M
CONKLIN,DANIEL L
CONNERS,RONALD E
COVINGTON,RALPH L
CRAGUN,DONALD W
CRAGUN,JOAN
CRAIG,N E
CRANE,B A
CRISTOFOR,EUGENE
<CRUME,L L
CHUPI,JOSEPH A
DAVIDSON,CHARLES LEWIS
DAVIS,JEFFREY B
DAVIS,R DREW
DAY,F W
DE FAZIO,M J
DE GRAAF,D A
DEAN,JEFFREY S
DENNY,MICHAEL S
DEVLIN,SUSAN J
DINEEN,THOMAS J
DOZDLINE,BARBARA ANN
DOLOTTA,T K
DOMANSKI,BERNARD
DOWDEN,DOUGLAS C
DRAKE,LILLIAN
DUDLEY,ELIZABETH H
DUFFY,F P
DUGGER,DONALD D
DUMAIS,VALERIE
DWYER,T J
EDELSON,DAVID
EHLINGER,JAMES C
EISEN,STEVEN A
<ZITZELBACH,DAVID L
ELY,T C
ELZER,RICHARD L
EMERSON,DOMINIQUE M

COVER SHEET ONLY TO

EPSTEIN,ROBERT H
ESSERMAN,ALAN R
FABISCH,M P
FABRICIUS,WAYNE N
FAGAN,EDWARD C
FAHRMANN,VIRGINIA L
FAIRCHILD,DAVID L
FASNACHT,MARK STEVEN
FEDER,J
FELDMAN,STUART I
FISCHER,HERBERT B
FISHMAN,DANIEL H
FLANDRENA,R
FLOZER,ELLEN F
FLYNN,JOHN F
FOO,YEOW PIN
FORTNEY,F J
FOUNTOUKIDIS,A
FOY,J C
FRANKLIN,C M
FRANKLIN,JAMES W
FRANK,AMALIE J
FRANK,RUDOLPH J
FRASER,A G
FREEMAN,K G
FREEMAN,MARTIN
FREENY,STAN L
FREMON,R C
FRENCH,A F,JR
FREY,H C
FRIED,LAWRENCE K
FUCHS,EDWARD
GABBE,JOHN D
GANA,JORGE L
GASPERINI,R T
GEARY,M J
GEHANI,NARAIN H
GELBER,CHERON L
GEPNER,JAMES R
GEWIRTZ,WILLIAM L
GEYLING,F T
GIBB,KENNETH R
GIBSON,T A
GIMPEL,J F
GITHENS,J A
GITHENS,JAY L
GLAZER,STANLEY
GLUCK,P G
GLUSHKO,ROBERT J
GOLABEK,RUTH T
GOLDSTEIN,A JAY
GORMAN,J E
GORTON,D R
GRAHAM,R L
GREENWALGH,H WAIN
GREENWOOD,I S
GROPP,R H
GRUENWALD,JOHN
GUIDI,PIER V
GUYTON,ARTHUR
HAFER,Z H
HAGELBARGER,DAVID

COVER SHEET ONLY TO

HAHN,ELIZABETH A
HAISCH,R F,JR
HALFIN,S
HALL,ANDREW D,JR
HALL,M A
<HALL,MILTON S,JR
HALL,WILLIAM G
HALPIN,T
HAMILTON,LINDA L
HAMILTON,PATRICIA A
HARRINGTON,T DORSEY
HARRIS,F L
HARRIS,MARION C
HARTMANN,R H
HARUTA,K
HAWKINS,DONALD T
HECHT,MATTHEW S
HIGINBOTHEM,F A
HOCHBERG,GLENN A
HOPPERT,D J
HORGAN,BOB
HOUGHTON,THOMAS F
HOWARD,L A,JR
HOWARD,PHYLLIS A
HO,DON T
HO,JENNY
HO,TIEN-LIN
HSU,TAU
HUNNICUTT,C F
IPPOLITI,G J
IRVEN,J H
IRVINE,A M
JACKOWSKI,S J
JACKSON,H M,2ND
JACOBS,H S
JOHNSON,STEPHEN C
JOHNSTONE,I
JOHNSTON,EUGENE E
JONES,JACQUELYN H
JONES,JOSEPH E
KACHURAK,J J
KADAKIA,SHIRISH S
KAEN,MORRIS S
KANE,JOHN M
KAPLAN,FRANK
KAPLAN,MICHAEL M
KARR,M A
KARTUS,JESSE S
KAUFMAN,ANN S
KAYEL,R G
KEESE,W M
KELLY,L J
KENNIGHAN,BRIAN W
KERR,F X,JR
KERTZ,D B
KESSELL,D L
KICHURA,J A
KILLMER,JOHN C,JR
KINKER,JEFFERY R
KLAPMAN,RICHARD M
KLEINER,R T
KLEIN,ROBERT M
...
512 TOTAL

* NAMES BY AUTHOR > CITED AS REFERENCE < REQUESTED BY READER (NAMES WITHOUT PREFIX WERE SELECTED USING THE AUTHOR'S SUBJECT OR ORGANIZATIONAL SPECIFICATION AS GIVEN BELOW)

MERCURY SPECIFICATION.....

COMPLETE MEMO TO:
542-DPH 5424-SUP

COVER SHEET TO:
COSAIR = COMPUTING/SPECIAL APPLICATIONS/INFORMATION RETRIEVAL

TO GET A COMPLETE COPY:

1. BE SURE YOUR CORRECT ADDRESS IS GIVEN ON THE OTHER SIDE.
2. FOLD THIS SHEET IN HALF WITH THIS SIDE OUT AND STAPLE.
3. CIRCLE THE ADDRESS AT RIGHT. USE NO ENVELOPE.
4. INDICATE WHETHER MICROFICHE OR PAPER IS DESIRED.

NO CORRESPONDENCE FILES
MC MA127

TM-80-5424-1
TOTAL PAGES 29

PLEASE SEND A COMPLETE

( ) MICROFICHE COPY ( ) PAPER COPY

TO THE ADDRESS SHOWN ON THE OTHER SIDE.

Bell Laboratories

subject: Shell Commands for Relational Operations on Fielded Files (SCROFF) Case 40016-100

date: January 2, 1980

from: S. M. Salchenberger
5424-800102.01TM

TM-80-5424-1

MEMORANDUM FOR FILE

1. INTRODUCTION

The need for data base management systems to control and provide access to shared data has long been recognized. The relational model [1] and systems based on it provide especially convenient tools for managing that data.

Individual users of computer systems have data management requirements as well. While desiring the power and flexibility of a high level interface to their own data, they do not want to relinquish control over it. In particular, they want to be able to use any facilities available on the local system (including, for example, text editors) to manipulate their files. These needs will become even more evident as increasing numbers of unsophisticated users gain access to information management related services, such as Home Information Systems [3].

The UNIX* operating system provides a wide variety of commands to manipulate files and retrieve data from them, including awk, comm, diff, grep, join, and others [9]. However, these commands are not integrated in any manner, do not permit the user to reference columns of a fielded file** by name rather than number, and do not together constitute a relationally complete [2] and convenient to use set of tools.

This memorandum describes new commands (SCROFF) available on UNIX which are meant to bridge the gap between existing commands and a complete, integrated set of data management functions. It is felt that these tools will be useful to the individual UNIX user, but may also fill many project related needs such as software administration. In addition, they may provide the starting point for further work concerning customer tools required by new information management based network services.

---

* UNIX is a trademark of Bell Laboratories.

** A fielded file will be taken as a file of ASCII characters to be interpreted as one record per line. Each record contains one or more fields (or columns) which are delimited by a unique character (usually tab or space).

SCROFF commands are loosely based on relational algebra [4], but regular expressions are permitted in qualifications, increasing its selectivity power. SCROFF also supports set operations for which no relational algebra analogy exists. Invoked at the shell level, they permit operations on any user owned fielded files and permit fields in those files to be referenced by name rather than number.

2. DESIGN

2.1 Overview

The intent of SCROFF is to supplement, not replace current UNIX facilities. The major function performed by the system is to translate relational algebra expressions into appropriate shell level UNIX commands, and then to execute those commands. In particular, the UNIX commands awk, sort, and comm are invoked heavily by SCROFF. The system does not recognize data types and supports variable length fields and records.

As the translation function mentioned above was designed, the need for two additional utilities became evident. These include a multi-field relational equi-join and a formatter to generate tabular output from fielded input files. Separate utility functions with a syntax compatible with conventional shell commands provide these capabilities.

Since most of the real work involved in satisfying queries and updates is performed by existing UNIX tools, the system is small and was designed and implemented in about two man months, including the work involved in the utility functions.

2.2 The Command Translator

The command translator is a single UNIX process which parses its input and generates a file of shell commands to perform the required operations. The lexical analysis and parsing functions are performed by programs generated by LEX and YACC, and the size of the translator is about 750 source lines, not including the LEX and YACC rules.

Any SCROFF command typed by a user is passed to the command translator, which then writes a file of shell (and utility function) commands. This compiled version of the command is then executed. SCROFF commands can be used anywhere other shell commands are, and in particular can be invoked through other shell procedures with parameter substitution capability. In addition, a limited facility (no parameters are allowed) exists for translating a sequence of SCROFF commands into shell script, and then saving the compiled version for future use.

2.3 Utilities

The JOIN utility provides a multi-field relational equi-join of its two fielded input files and writes its result to standard output as another fielded file. It assumes that both files are sorted in ascending order on the field or fields to be joined over.

The PRINT utility takes as input a fielded file and generates a paginated tabular listing of that file with user controllable column widths. Fields which overflow their specified (or defaulted) width are normally folded to the next line (as often as necessary), but can be truncated if desired.

Although these functions were designed for SCROFF, they can be invoked directly with any fielded file as input. Manual pages for these commands may be found in Appendix B.

3. USE OF SCROFF

3.1 Getting Started

All SCROFF commands reside in the directory

/b2/sams/SCROFF/bin

on the UNIX-B machine. Any user planning to invoke SCROFF on a regular basis should include this directory in his PATH variable (but see the Appendix A entry for the join command).

In addition, an initialization procedure (SCROFF.init) must be executed before any other SCROFF commands can be invoked. This will create a directory called "system" in the current directory which is used for various (mostly temporary) files needed by SCROFF during command translation and execution. See Appendix A for a manual page.

All files accessed by SCROFF must reside in the current directory when invoking SCROFF commands. This permits the user to maintain multiple data bases in separate directories. However, a SCROFF.init must be performed from each directory in which SCROFF relations will reside.

SCROFF requires the same field delimiter character in all files (in a given data base). The tab character is the default, but this can be changed at compilation time. Null values can be indicated by consecutive delimiters. This restriction, together with the convention that each line represents one record (tuple), are the only formatting requirements the user must follow in preparing a file to be accessed by SCROFF.

To provide SCROFF with the correspondence between field positions and field names within a file (to be thought of as a SCROFF relation), the user invokes a create command. For each such file, SCROFF creates a descriptor file with the field names. If the name of the relation is filename, then the descriptor file is named h.filename. The descriptor file allows SCROFF to relate field names with their position in the file when subsequent commands are invoked, and it also permits renaming and/or logical reordering fields within a file via the alias command (to be discussed).

3.2 SCROFF Commands

The set of commands currently supported by SCROFF is the following:

1. SCROFF.init - establishes a "system" directory which permits other SCROFF commands to be executed from the current directory (but does not alter the user's PATH variable).

2. alias - renames and/or logically reorders the column names of an existing relation, permitting it to be referenced by another (aliased) name.

3. append - appends tuples of one relation to another.

4. create - defines a new relation and (optionally) initializes it to a user specified input file.

5. delete - removes tuples from a relation.

6. destroy - removes a relation (in the case of an alias, it removes only the descriptor file).

7. join - performs a relational equi-join over all common fields (alias can be useful in causing join to operate over desired fields).

8. print - produces a formatted print of a relation.

9. project - restricts a relation to specified fields.

10. save - permits a file of SCROFF commands to be translated and saved for later use (see discussion below).

11. select - the relational algebra select function, except that regular expressions may be specified.

12. setdiff - forms the set difference of two relations, which must have identical domain names (but not orders). Again, alias can be useful prior to a setdiff.

13. setunion - performs a set union of two relations and can leave duplicates.

14. unique - sorts a relation and removes any duplicates.

Complete descriptions of these commands can be found in Appendix A, and Appendix C contains a short example using many of them.

As was mentioned in the above list, the save command permits the user to translate a set of commands once and save the results for later use. This eliminates some overhead each time the commands are executed, but the facility (as currently implemented) does not permit parameter substitution within the set of commands to be saved. If this is required, the user may establish a shell procedure with the SCROFF commands themselves inside, thereby obtaining the complete power of the shell.

4. SCROFF AND OTHER SYSTEMS

SCROFF was designed as an integrated set of data management tools, not as a DBMS. Therefore, it does not possess such features as concurrency control, transaction control, or crash recovery. However, it provides a more than relationally complete set of commands and an easy to use syntax. It also supports null values and variable length fields and records.

While most of the power of SCROFF can be attributed to its underlying tools (awk, comm, sort, JOIN), its performance is limited by the fact that executing a command involves creating and interpreting a shell command file. Therefore, its fixed overhead causes some delay in processing small files, but this is not noticeable on larger files. The following table illustrates the performance for a simple select statement (specifying equality on one field) and for a join operation. All the relations involved in the tests contained approximately 25 bytes per tuple.

operation	relation size (tuples)	result size (tuples)	user time (seconds)	system time (seconds)
select	2112	24	5.6	3.5
select	1	1	0.3	2.4
join	1026 x 1050	2728	28.3	15.0
join	1 x 1	1	0.4	4.0

The following measurements are for the same operations, but for the case where the queries have been "pre-compiled" via the save command. As may be seen, the savings gained by employing save is a substantial portion of the total time only for queries on very small tables.

operation	relation size (tuples)	result size (tuples)	user time (seconds)	system time (seconds)
select	2112	24	5.6	2.7
select	1	1	0.1	1.2
join	1026 x 1050	2728	28.5	11.7
join	1 x 1	1	0.1	2.0

There are four* other relationally based data management systems of which the author is aware which run on UNIX: UNITY [6], INGRES [8], DMC [7,11], and POLARIS.

4.1 INGRES, DMC, and POLARIS

These systems differ from SCROFF in a number of respects. INGRES and DMC do not support shell level interaction (the author's knowledge of POLARIS is limited in this respect), and none of these supports variable length fields or offers the capability of specifying regular expressions in the select and delete statements. In addition, they maintain control over the user's files due to their underlying access methods. On the other hand, all three provide direct interfaces to the system through C language programs. The greatest similarity between SCROFF and any of these systems is the language, which closely resembles that of DMC.

4.2 UNITY

Because of its simplicity and shell level interface, UNITY is more closely related to SCROFF than either of the other two systems. However, a number of important distinctions exist. In general, SCROFF offers more query power, while UNITY has more features which make it more like a true DBMS and less like a set of related tools.

SCROFF permits arbitrary Boolean conditions in the select statement, while UNITY is restricted to a simple conjunction or disjunction. This distinction carries over to the delete statement as well. Further, UNITY does not permit regular expressions in select or delete and does not have a setdiff command. This fact, coupled with a lack of a relational algebra division function, makes UNITY less than relationally complete. UNITY also restricts the join operator to one common field in the two source relations.

On the other hand UNITY provides many features not present in SCROFF, including indices into files, limited concurrency control, and access to files on remote machines. Statistical functions and a replace (or alter) command are also available on

* LAMBDA-R [5] is no longer supported.

UNITY, but not SCROFF (however, see Section 5). While these features may be desirable, some of them, including indices and concurrency control, tend to give the system rather than the user control over the data.

5. EXTENSIONS AND CONCLUSIONS

Several enhancements to the current set of SCROFF capabilities are contemplated. These include the implementation of statistical functions, a replace command, an interface to a report generator, and interfaces to encrypted files.

As a set of tools for managing personal data, SCROFF fills a gap between existing commands on UNIX and a complete DBMS. It provides more than a relationally complete interface to user files, while permitting the user to retain control and ownership over those files.

6. ACKNOWLEDGEMENT

The concept of much of this work came from an idea based on an unpublished memorandum by R. L. Martin [10].

S. M. Salchenberger

IH-5424-SMS-nroff

Atts.
References
Appendix A (p.1 - 16)
Appendix B (p.1 - 4)
Appendix C

REFERENCES

[1] E. F. Codd, "A Relational Model of Data for Large Shared Data Banks", CACM 13, 6, June, 1970.

[2] E. F. Codd, "Relational Completeness of Data Base Sublanguages", Courant Computer Science Symposium 6, Data Base Systems.

[3] J. J. Carroll, "Area 50 Planning for Home Information Services (HIS)", 5422-780815.01CN.

[4] C. J. Date, An Introduction to Data Base Systems, 2nd Edition, Addison Wesley, 1977.

[5] Lambda-R User's Manual, edited by Software Technology Laboratory, Jan. 1979.

[6] P. D. Ting and D. C. Tsichritzis, "A Micro-DBMS for a Distributed Data Base", TM-79-1712-2.

[7] W. L. Gewirtz, R. A. Medhekar, F. K. Ng, A. L. Robertson, and S. M. Salchenberger, "A Relational Data Management System for Electronic Switching Systems", Proceedings of COMPSAC 1979.

[8] M. Stonebraker, E. Wong, P. Kreps, and G. Held, "The Design and Implementation of INGRES", ACM Transactions on Database Systems 1, No. 3, September, 1976.

[9] PWB/UNIX User's Manual Release 2.0, edited by M. E. Pearlman, et. al., June, 1979.

[10] R. L. Martin, "A Simple, Table-Driven Report Compiler", unpublished memorandum.

[11] F. K. Ng, "A High Level (Relational) Interactive Interface to the DMC", 5424-790911.02MF.

This appendix contains manual pages for each of the SCROFF commands. Required words in the command syntax are underlined, and optional words or clauses are enclosed in square braces. The words "rel1" "dom1" (and deviations on them) are used to represent generic relation and domain names, respectively. In general, these are restricted to consist of alphanumerics, periods, and underscores, and relation names cannot be longer than 14 characters in length (unique on the first 12).

All commands described in this appendix reside in the directory

/b2/sams/SCROFF/bin

on the UNIX-B machine.

NAME

SCROFF.init - initialize a directory to permit SCROFF commands

SYNOPSIS

SCROFF.init

DESCRIPTION

SCROFF.init creates a directory called "system" in the current directory (and creates several directories under the "system" directory) whose existence is required prior to the execution of any other SCROFF commands.

NAME

alias - reorder and/or rename the domains of a relation

SYNOPSIS

alias rel1 to rel2 [doma to newdoma [domb to newdomb]...]

DESCRIPTION

Alias provides an alternate name (rel2) for the given relation (rel1) and permits reordering and/or renaming of the domains. In the synopsis above, the aliased relation rel2 would possess domain names newdoma, newdomb, ... in that order. Any domains from rel1 which are not specified are put at the end of the aliased relation's domain list in the order in which they appear in rel1 (with the same names).

A descriptor file for the alias is created. The relation name and physical domain orders referenced in the descriptor file are those from the source relation (see create(A)).

SEE ALSO

create(A), destroy(A), join(A), setdiff(A), setunion(A)

BUGS

The relation can have a maximum of 100 domains.

NAME

append - add tuples to a relation

SYNOPSIS

append rel1 to rel2

DESCRIPTION

Append places all tuples in rel1 into rel2. Rel1 is unaffected. If duplicate tuples result, they are not removed. If removal of duplicates is desired, a unique operation must also be performed. The two relations must have the same number and names of domains, although permutations are allowed (the alias operator can be helpful in the second respect). The only difference between append and setunion is the destination of the result.

SEE ALSO

alias(A), setdiff(A), unique(A)

BUGS

Although the descriptor files are checked for compatibility, the relations themselves are not verified to have the same number of domains.

NAME

create - create a relation

SYNOPSIS

create rel1 with domains dom1 [dom2 ...] [from srcefile]

DESCRIPTION

Create establishes a new relation with the named domains in the named order. A descriptor file for the relation (h.rel1) is created. If the relation has n domains, then the descriptor file will have n+2 lines. The first line contains the name of the relation, the second contains the number of domains (n), and the remaining lines contain the domain names, a tab, and the physical order of the domain (1, 2, etc.) in the file. If no srcefile is specified, the relation file rel1 is created and left empty. If the srcefile is rel1, it is left untouched. If the srcefile is different than rel1, it is copied to rel1 and left uncharged.

SEE ALSO

alias(A), destroy(A)

BUGS

The relation can have a maximum of 100 domains.

Srcefile is not checked for compatibility with the definition of the relation.

NAME

delete - delete tuples from a relation

SYNOPSIS

delete from rel1 where condition
delete rel2 from rel1

DESCRIPTION

Delete removes tuples from the relation rel1. In the first form, the tuples removed are specified by the condition which obeys the same rules as the condition in the select command.

In the second form, all tuples from rel2 which are present in rel1 are removed. The two relations must have the same domain names and the same number of domains, but permutations are allowed.

SEE ALSO

alias(A), select(A), setdiff(A)

BUGS

In the second form, the descriptor files are checked for compatibility. However, the relations themselves are not checked to verify that they have the same number of domains.

NAME

destroy - remove a relation and its descriptor file

SYNOPSIS

destroy rel1

DESCRIPTION

Destroy removes the descriptor file for the named relation. If the relation is not an alias, the relation itself is also removed, along with any descriptor files of relations which were aliased to it. If the relation is an alias, only its descriptor file is removed.

SEE ALSO

alias(A), create(A)

NAME

join - form the relational (equi-) join of two relations

SYNOPSIS

join rel1 rel2 [project on dom1 ...] [into rel3]

DESCRIPTION

Join forms the relational equi-join of the two named relations. The join is performed over all domain names which are common to the two relations (alias can be helpful here). If no domains are held in common, then the Cartesian product is formed.

If rel3 is specified, then that relation is created (as a SCROFF relation) and populated with the results of the join. Otherwise, the results are written to standard output.

The schema of the result consists of all domains in rel1 followed by all domains of rel2 which are not present in rel1. Note that this command has the same name as a standard UNIX command. The user's PATH variable must be set properly to provide access to this, rather than the standard UNIX join.

SEE ALSO

alias(A), JOIN(B)

BUGS

The result relation can have a maximum of 100 domains and 512 bytes per tuple.

NAME

print - print all tuples in a relation

SYNOPSIS

print rel1

DESCRIPTION

Print produces a formatted, tabular, paginated (66 lines per page) listing of the named relation on standard output. Column headings are supplied. More flexibility can be obtained by direct use of the utility function PRINT. If only specified rows or columns are desired, then a project and/or select should be performed.

SEE ALSO

PRINT(B), project(A), select(A)

BUGS

No tuple can have more than 512 bytes.

There must be the same number of domains in each tuple (null values can be indicated by consecutive delimiters).

NAME

project - select specified domains from a relation

SYNOPSIS

project rel1 on [dom1 ...] [into rel2]

DESCRIPTION

Project selects the specified domains from rel1 and prints the results (which may include duplicates) on standard output, unless rel2 is specified. In this case, rel2 is created (as a SCROFF relation) and populated with the results of the project. Note that project is logically equivalent to a select statement which specifies domains but not a condition.

SEE ALSO

select(A)

NAME

save - save commands for future use

SYNOPSIS

save

DESCRIPTION

Save reads SCROFF commands from standard input, translates them to appropriate shell script, and writes the results to standard output. Hence, this command is usually employed by redirecting both input and output. No create, alias, or destroy commands are permitted, and no parameter passing is possible. Any relations which are created (as the result of "into" clauses) are considered temporary. They will be created and destroyed each time the translated version of the command is run.

SEE ALSO

append(A), delete(A), join(A), print(A), select(A), setdiff(A), setunion(A), unique(A)

NAME

select - select rows (and columns) from a relation

SYNOPSIS

select [dom1 ...] from rel1 [into rel2] [where condition]

DESCRIPTION

Select extracts tuples from the relation rel1 according to the condition specified. If no domains are specified, all domains (fields) from the relation are selected. Otherwise, only those domains chosen are selected. If rel2 is not specified, results are written to standard output. If rel2 is specified, it is created (as a SCROFF relation) and initialized to the result of the select.

If no condition is specified, all rows are selected. If specified, the condition may be an arbitrary Boolean conjunction or disjunction of simple expressions, which may be grouped together as desired with parentheses. In specifying the simple expressions and the Boolean conjuncts and disjuncts, collisions with special shell characters are common. Therefore, SCROFF provides alternatives to avoid these difficulties. In the following discussion, it is assumed that all special characters (see sh(1)) will be escaped (with a backslash or by quoting) so they are passed properly to SCROFF.

A Boolean conjunct is specified by and or && and a disjunct by || or or. The valid simple expressions are summarized by the following:

```
domain_name relop value
domain_name regop regexp
domain_name relop domain_name
```

Domain name must be a valid name of a domain in the relation. Relop is one of the following:

= == .eq.	(equals)
!= .ne.	(not equal)
> .gt.	(greater than)
< .lt.	(less than)
>= .ge.	(greater than or equal)
<= .le.	(less than or equal)

Value can be any string, but if it contains characters other than alphanumeric, periods, and underscores, it must be quoted (and quotes must be escaped). Quotes within a string must also be escaped.

Regop is either ~ (for matches) or !~ (for doesn't match).

SCROFF recognizes any string delimited by slashes as a regexp, but the syntax of the regular expression must agree with that of awk(1).

SEE ALSO

awk(1), project(A)

BUGS

The relation can have a maximum of 100 domains and 512 bytes per line.

Awk seems to have difficulty with domains where some values are numeric and some values are non-numeric. Spurious results may be generated.

NAME

setdiff - compute the set theoretic difference of two relations

SYNOPSIS

setdiff rel1 rel2 [project on dom1 ...] [into rel3]

DESCRIPTION

Setdiff forms the set theoretic difference of the two relations (rel1 - rel2), optionally projecting the result on the specified domains. If no destination relation is specified, results are written to standard output. Otherwise, the new relation is created and initialized to the results of the operation. The new relation must not already exist. Both source relations must have identical numbers and names of domains, although permutations are allowed.

SEE ALSO

alias(A), delete(A)

BUGS

Although the descriptor files are checked for compatibility, the relations themselves are not verified to have the same number of domains.

NAME

setunion - compute the set theoretic union of two relations

SYNOPSIS

setunion rel1 rel2 [project on dom1 ...] [into rel3]

DESCRIPTION

Setunion forms the set theoretic union of the two relations, optionally projecting the result on the specified domains. Duplicates are not removed. If no destination relation is specified, results are written to standard output. Otherwise, the new relation is created and initialized to the results of the operation. The new relation must not already exist. Both source relations must have identical numbers and names of domains, although permutations are allowed.

SEE ALSO

alias(A), append(A), unique(A)

BUGS

Although the descriptor files are checked for compatibility, the relations themselves are not verified to have the same number of domains.

NAME

unique - remove duplicates from a relation

SYNOPSIS

unique rel1

DESCRIPTION

Unique sorts the named relation in ascending order and removes all duplicate tuples. The collating sequence is based on the physical positions of the domains in the relation, so in the case of an aliased relation, a unique followed by a print will not necessarily produce a listing sorted on the first domain.

This appendix contains manual pages for the utility functions PRINT and JOIN. Although these were designed for use in conjunction with other SCROFF commands, their syntax is compatible with other shell commands, and they can be used on any fielded file.

Both commands described in this appendix reside in the directory

/b2/sams/SCROFF/bin

on the UNIX-B machine.

NAME

JOIN - relation algebra (multi-field) equi-join

SYNOPSIS

JOIN [options] file1 file2

DESCRIPTION

JOIN forms on standard output the equi-join of the two specified input files, each of which must be sorted in ascending ASCII sequence on the (concatenation of) the fields over which they are to be joined. If the first file named is "-", standard input is read. Fields to be joined over are specified by the "-1" option (see below). If no fields are specified, the Cartesian product of the two files is formed. If no projection is specified (see below), the output consists of one line for each matching line of file1 and file2. Fields in file1 are displayed first, followed by any fields in file2 which were not joined over. JOIN recognizes the following options:

-1list List is a sequence of integers representing field numbers and separated by commas. It must be passed as a single argument. If one such option is present, it specifies the field numbers in both files over which to join. If two such options are present, the first specifies the joined fields for the first input file, and the second specifies the joined fields for the second input file.

-plist List is a sequence of pairs of integers of the form m.n separated by commas. It must be passed as a single argument. For each pair m.n in the list, m specifies a file number (1 or 2) and n specifies a field number to be printed. If this option is not specified the fields to be printed are defaulted as indicated above.

-sc The character c is used instead of the default (tab) field separator, both on input and output.

SEE ALSO

join(A), join(1)

BUGS

Input lines are restricted to a maximum of 100 fields and 512 characters.

Output lines are restricted to a maximum of 100 fields.

If the input files are not sorted, erroneous results will be produced.

NAME

PRINT - produce tabular output from a fielded file

SYNOPSIS

PRINT [options] filename [headerfile]

DESCRIPTION

PRINT produces a formatted tabular output of the named file (filename). The input file is assumed to consist of lines, each of which contains (the same number of) fields. A field is any sequence of characters separated from adjoining fields by a special delimiter character (default is tab). If headerfile is specified, it is assumed to be a SCROFF descriptor file containing column names for the fields in filename. In this case these names will be written as column headings on each page of the output. Otherwise, no column headings will appear. The format of the output is controlled by the following options:

-Bnum Set the bottom margin on each page to num lines. Default is 5. Note that some pages may contain fewer blank lines at the bottom (see BUGS below).

-Tnum Set the top margin of each page to num lines (plus a line for the heading and page number - if any). Half the blank lines will be above the header line and half below. If this number is odd, it is rounded to the next higher even integer. Default is 4.

-f Don't place blank lines at the end of the last page to make the last page the specified page length (see the -l option). Default is to blank fill each page including the last to the specified length.

-hhead Use the string head as a page heading for each page. The header string must be passed as a single argument. The heading follows the page number unless the -n option is specified.

-lnum Num is a number to be used as the number of lines per page. Default is 66, and this number must be at least 6 greater than the sum of the top and bottom margins (see the -B and -T options). A very large value for num coupled with the -f option will produce a listing of the relation similar to a cat(1) as opposed to a pr(1).

-mnum Num is a number to be used as the maximum width to which a column will be expanded before permitting folding or truncation. Default is 20. This maximum can be overridden on a field by field basis by use of the -w option.

-n Do not number pages. This does not affect the -h option, if specified.

-oc Use the character c as the field delimiter on output. Default is "|".

-plist List is a list of integers separated by commas which specifies the field positions to be printed and the order in which to print them. It must be passed as a single argument. Default is to print all fields in sequential order.

-sc Use the character c as the field delimiter on input.

-t If columns overflow their allotted width (see the -w option), truncate them. The default is to overflow to as many lines as is necessary.

-wlist List is a list of integers separated by commas which specifies the width of the columns for the fields to be printed. It must be passed as a single argument. The first number applies to the first field to be printed, and so on. These numbers override the default for the maximum allowable field width, which is either 20 or is specified by a -m option. If this option is not specified, each column width is set to the maximum allowable field width or to the minimum width necessary to permit all values of that field to be printed without folding, whichever is smaller. Individual values of 0 in the list will be ignored, permitting the user to override the default field widths for some, and not all fields.

BUGS

The input file is restricted to 100 fields per line and 512 characters per line.

If the last record on a page causes folding to multiple lines, the number of blank lines at the bottom of that page will be reduced. In the extreme case, that page may be longer than the rest and have no blank lines at the bottom.

One spurious page with just a heading and no records is sometimes produced.

This sample set of commands makes a relation from a file and resolves the question of "Who makes more than his or her boss?".

```
? create employees with domains name salary supervisor from
  employees
? unique employees
? print employees
```

Page 1 employees relation

name	salary	supervisor
Alberts	40000	Ross
Brooks	50000	Alberts
Johnson	40000	Alberts
Jones	85000	Roberts
Olson	70000	Roberts
Roberts	80000	Smith
Ross	130000	Smith
Schwarz	50000	Ross
Smith	120000	
Williams	35000	Ross
Wilson	70000	Jones

```
? alias employees to aliasemp name to supervisor salary to
  sup_salary supervisor to dont_care
? join employees aliasemp project on name salary supervisor
  sup_salary into temprel
? select name supervisor from temprel where salary .gt. sup_salary
```

name	supervisor
Brooks	Alberts
Jones	Roberts
Ross	Smith

```
? destroy temprel
```