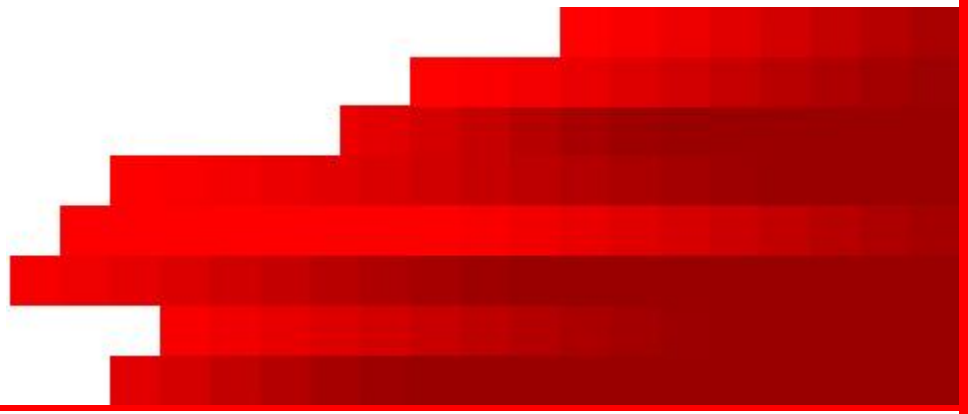
# UNIX and Shell scripting

## Module 2 – File Commands and Filters



HSBC Technology and Services

HSBC
The world's local bank

# Objectives

- Search for a file using '*find*' command
- which command
- File comparison commands
- File redirection
- Pipes and Filters

# Locating files

- find - Searches the named directory and it's sub-directories for files with a matching expression.

Example:

- $ find ./ -name “employee.txt"
  - Which searches the current directory ( and all of its sub-directories ) for a file employee.txt and prints them out.
- Options
  - -type
    -type f → search for ordinary files
    -type d → search for directory
  - -name – name of the file or a directory
    Pattern searching – always string enclosed in quotes

# Locating files (contd..)

## -mtime

- -mtime n → File modified in exactly n days
- -mtime +n → File modified in more than n days
- -mtime –n → File modified in less than n days

## -exec

- This option allows to execute a command for files located by find command
- $ find . –name temp –type f –exec rm {} \;

  This will remove temp files

## -ok

- Seeks user's confirmation for command execution of –exec option
- $ find . –name temp –type f –ok rm {} \;

# Input / output redirection

- Input redirection
- Output redirection
- Standard error redirection

- 0 standard input
- 1 standard output
- 2 standard error

# Input redirection

- Changing the default input source
- Input redirection operator ‘<‘
- $ command < filename

- Example :
  - $ cat <employee.dat

# Output redirection

- Changing the default destination of the output
- The output redirection operator is ‘ >’
- $ command > filename

- Example:
  - $ cat emp.dat > emp.out
  - $ date > todays_date
  - $ cat > file1
  - $ cat >> file2

# Error redirection

- Changing the default input source
- Input redirection operator ‘2 >‘
- $ command 2 > errorfile
- Eg :
  - $ cat customer.dat > customerdetails.txt  2 > errorfile

# Pipes and filters

- Pipes :Connects two or more commands
- Filters : Unix command which takes its input from standard input file, processes it and sends the output to standard output file
- In this lesson, we are going to look at the following filters:

  - wc
  - pg
  - more
  - tee
  - tr
  - grep
  - sort
  - cut
  - paste
  - head
  - tail

# Pipes

- The pipe ('|') operator is used to create concurrently executing processes that pass data directly to one another.
- It is useful for combining the commands to perform more complex functions.

- Example: $ who | grep “Raghu”

# Filters: grep

- grep – ( global search for regular expression and print)
- grep is the standard searching and selection utility

- $ grep [options] “pattern” <filename>

- Options :
  - -n : prints the line along with line numbers
  - -v : the reverse search criterion
  - -c : display only a count of matching patterns

# Filters: grep (cont..)

## Examples:

- grep “Raghu” employee.dat
  - Prints the line(s) containing the string Raghu.
- grep –n “Raghu” employee.dat
  - Prints the lines containing the string Raghu preceded by the line number
- grep –v “Raghu” employee.dat
  - Display the lines of the file excluding the lines containing the string Raghu.

# Regular expressions

| Regular expressions | Description |
|---|---|
| [ ] | To specify a pattern which consists of any one character. “[xyz” specifies the pattern either “x” or “y” or “z” |
| [ ] with hyphen | Range of characters “[1-9]” indicates a number in a range of 1 to 9 |
| ^ | The pattern following it must occur at the beginning of the line.<br>“^[123]” indicates the line must start with 1 or 2 or 3. |
| [^] | Searched string should not contain the characters followed by the ^. |

# Regular expressions (cont..)

| | |
|---|---|
| . | Single character, “[123]”. Indicates that the pattern should have 1,2 or 3 followed by single character. |
| $ | Specifies the pattern preceded should appear at the end of each line.“[abc]$” indicates a, b or c must appear at the end of the line. |

# Use of grep as a filter

Example:

- To find if employee.dat is present in the current folder

```
ls | grep “employee.dat”
```

- To find if a given user is logged in or not

```
who | grep “Raghu”
```

# sort

- Sort
  - Sorts the contents of a file.
    - `sort [-b f n r o t] [file name(s)]`
  - Takes the contents of a file(s) and displays it in sorted order.
  - Flags:

| | |
|---|---|
| -b | ignores blanks |
| -f | change all lower case letter to upper case before comparing |
| -n | numeric sort |
| -r | reverse usual order |
| -o | sends output of command to some file |
| -t | field delimiter |

  - E.g. To sort the Emp.txt on 2nd field in reverse order

```
$sort -t, -n -r +2 Emp.txt
```

# sort (contd..)

- sort  <enter>
- Eg:
  - To sort the contents of a file
  - $ sort namelist
  - $ sort –r namelist
  - $ sort –n numbers ( to sort the numbers)
  - $ sort +2  -3  employee
  - $ sort +2 -3 +3 -4 employee
  - $ sort –t “:” +2 -3 –o newfile customer

# Using cut as a filter

- To display a sorted list of the first name and salary of all employees belonging to Sales.

```
$ grep "sales" employee | cut –d " "-f 2,8 | sort
```

# cut

- $ cut –c4-9 employee
  - Display characters 4 to 9 from each record
- $ cut –d " " –f2-4 employee
  - Display 2-4 fields
- $ cut –f 4- employee
  - Display 4th field onwards

# paste

- Merges files horizontally.

  $ paste <file1> <file2>

  Options –d delimiter

# Examining the file contents

- Head[-n]  display first n lines of a file
- tail[+/-n]
- wc[-lcw]
- more
- tr
- tee

# head and tail

- head employee will display first 10 lines
- head -5 employee
- tail employee last 10 lines from the employee
- tail -7 employee
- tail +10 employee display lines from 10th till the end of the file.

# wc

- wc  word count

Options:

-c  - character count

-w  - word count

-l  - line count

- $ wc employee

- Examples:
  - To count the number of files in a directory
  - To count the number of users currently logged in

# more

- Command displays 23 lines at a time
- ls –l | more
- Display the directories one screenful at a time

# tee

- Used to redirect the output to two different destinations
- Transfers data without any changes to the destination

- Example:
- ls –l | grep “^-” | tee filelist |  tr –s “ “ | cut –d “ “ –f 5,9

# tr

- Translates the input into some other form
- Squeezes the repetitive characters from the input

- tr –s “ ” <  <filename>

Examples:

- Convert all lowercase to uppercase
  - cat customer.dat | tr “[a-z]” “[A-Z]”
- To display only login names and time of login from the output of “who” command
  - who | tr –s “ ” | cut –d “ ” –f1, 5

# Hands on

- Display the frequencies of different words present in the input
  -
- To find the largest file in the current directory
  -


**About HSBC Technology and Services**

HSBC Technology and Services (HTS) is a pivotal part of the Group and seamlessly integrates technology platforms and operations with an aim to re-define customer experience and drive down unit cost of production. Its solutions connect people, devices and networks across the globe and combine domain expertise, process skills and technology to deliver unparalleled business value, thereby enabling HSBC to stay ahead of competition by addressing market changes quickly and developing profitable customer relationships.

**Presenter's Contact Details**:
Name: Lorem ipsum
Role: Lorem ipsum
Direct: +00 00 000
Email: loremipsum@hsbc.com

Name: Lorem ipsum
Role: Lorem ipsum
Direct: +00 00 000
Email: loremipsum@hsbc.com

Restricted for company use only