Assuming Windows CMD – should be better on Mac OSX and Linux

Create Folders in your storage space.

Important: all scripts and stored procedure need updating with your paths. Hint search for '…' and fill in!

Important: all loops, in scripts and stored procedures (e.g. cursors), need to be checked. (Most have their range restricted for testing purposes).

```
rem dataset folder

c:\> cd $YOUR_HOME_DIRECTORY$

C:\...> mkdir cbirCore10k

rem tessellation sub folders

C:\...\cbirCorel10k>mkdir t12_100wby100h

C:\...\cbirCorel10k>mkdir t4_200wby150h

C:\...\cbirCorel10k>mkdir t2_200wby300h

rem scripts folder

C:\...\cbirCorel10k>mkdir scripts
```

Get a library from a source (with curl or wget) (or ask JV)

Edit getall.bat (with below) and execute on CMD prompt.

```
echo off

rem window 10 script in cmd
rem jv dec 2106

rem depends on imagemagick installed (with legacy option if v >= 7)
rem copies 1000 jpeg from Corel dataset of low res photos

rem the for next loop IN (a,s,t) reads from a step s to t
for /L %%a in (1,1,1000) Do curl -O http://www.ci.gxnu.edu.cn/cbir/GHIM20/%%a.jpg
```

Generic image deatils

```
C:\...\cbirCorel10k>identify -verbose 34*.jpg

…
```

Calculate md5 (or something similar)

```
C:\...\cbirCorel10k>identify -format %# 345.jpg

e5e976652f22bc4a3d604663eda2ccf2896a503b5652ab446985a2255b02e377
```

Calculate file type, colour depth, size etc

```
C:\...\cbirCorel10k>identify -format "{\"filename\":%f, \"filetype\":%m, \"height\":%h, \"width\":
%w, \"hash\":%#, \"colourspace\":%[colorspace]} \n" 98*.jpg

{"filename":98.jpg, "filetype":JPEG, "height":300, "width": 400,
"hash":06e161148edffe16e405c2045a247ae06d67c73e17b747ac2d1961985db85d82,
"colorspace":sRGB}

{"filename":980.jpg, "filetype":JPEG, "height":300, "width": 400,
"hash":2834b4d3210b4515445b02a406b21b7b01817f4adb7907a93b50938b692c6301,
"colorspace":sRGB}

{"filename":981.jpg, "filetype":JPEG, "height":300, "width": 400,
"hash":9c359c33b3fee4e4491c3d994c34a8c7de7d40da4f491a587fc2671aab0b5aeb,
"colorspace":sRGB}

{"filename":982.jpg, "filetype":JPEG, "height":400, "width": 300,
"hash":87b7524fbeb37537469efc8fc3be34804c557a21ea6cf175f48647debdefb4a0,
"colorspace":sRGB}

{"filename":983.jpg, "filetype":JPEG, "height":300, "width": 400,
"hash":9befc5510938bd9b79e92908cc2260e11982dd96ad983b5b555eaed37563f552,
"colorspace":sRGB}
```

Compare two (2) images and output the level of difference (in windows) over image (ae absolute error -> pixel count) (ncc normalised ->0 to 1 with 1 being identical)

(note: fuzz qualifier is useful for jpeg files as *similar* colours are aggregated)

(note: compare is symmetric)

```
C:\...\cbirCorel10k>compare -fuzz 10% -metric ae 3.jpg 1.jpg -compare

43849

C:\...\cbirCorel10k>compare -fuzz 10% -metric ncc 1.jpg 3.jpg -compare

0.409322

C:\...\cbirCorel10k>convert -fuzz 10% -metric ae 1.jpg 3.jpg -compare -format "%[distortion]" info:
```

43849

Image tessellation example

c:\...\cbirCorel10k>**convert -quality 100 -crop 100x100 1.jpg t12_100wby100h/1_tile100by100_%02d.jpg**

Image portrait (where h>w) or landscape (where w>h) test

...\cbirCorel10k>**convert 1.jpg -format "%[fx:(w/h>1)?1:0]" info:**

1

...\cbirCorel10k>**convert 66.jpg -format "%[fx:(w/h>1)?1:0]" info:**

0

Image colour histogram (by 8 BLACK_BLUE_LIME_CYAN_RED_MAGENTA_YELLOW_WHITE)

C:\...\cbirCorel10k>**convert 1.jpg -format %c -depth 1 histogram:info:-**

```
    113316: (  0,  0,  0) #000000 black
        13: (  0,  0,255) #0000FF blue
         7: (  0,255,  0) #00FF00 lime
        25: (  0,255,255) #00FFFF cyan
      5563: (255,  0,  0) #FF0000 red
         5: (255,  0,255) #FF00FF magenta
       821: (255,255,  0) #FFFF00 yellow
       250: (255,255,255) #FFFFFF white
```

Script: uploadall.bat

List generic image details on dataset

```
@echo off
setlocal enableextensions enabledelayedexpansion

rem window 10 script in cmd
rem joseph vella dec 2106

rem depends on imagemagick installed (with legacy option if v >= 7)
rem depends on Corel 1000 low res images and named as 1.jpg, 2.jpg,
..., 1000.jpg

rem invoke scripts\uploadall.bat
rem invoke scripts\uploadall.bat > whatever.json


rem header line
rem echo CMD      FILE1 FILE2 AE    NCC


rem note the -format "% is -format "%% - ie %% is escaping for
literal %

FOR %%Z IN ("C:\...\cbirCorel10k\*.jpg") DO (
     rem echo %%Z
     identify -format "{\"filename\":\"%%f\", \"filetype\":\"%%m\",
\"height\":\"%%h\", \"width\":\"%%w\", \"hash\":\"%%#\",
\"colourspace\":\"%%[colorspace]\"} \n" %%Z
                                                                  )



rem post processing house keeping
exit /B

```

Script: histogram.bat

Gives a colour signature breakdown on eight colours.

Example run on a single file.

```
C:\...\cbirCorel10k\convert %%a.jpg -format %%c -depth 1 histogram:info:-
```

```
    113316: (  0,  0,  0) #000000 black
        13: (  0,  0,255) #0000FF blue
         7: (  0,255,  0) #00FF00 lime
        25: (  0,255,255) #00FFFF cyan
      5563: (255,  0,  0) #FF0000 red
         5: (255,  0,255) #FF00FF magenta
       821: (255,255,  0) #FFFF00 yellow
       250: (255,255,255) #FFFFFF white
```

```
echo off
setlocal enableextensions enabledelayedexpansion

rem window 10 script in cmd
rem jv dec 2106

rem depends on imagemagick installed (with legacy option if v >= 7)
rem depends on Corel 1000 low res images and named as 1.jpg, 2.jpg, ..., 1000.jpg


rem invoke scripts\histogram.bat
rem invoke scripts\histogram.bat > whatever.lst

rem Important read note at end of script to explain output

rem convert 1.jpg -format %c -depth 1 histogram:info:-


rem header line
echo CMD    FILE  BLACK_BLUE_LIME_CYAN_RED_MAGENTA_YELLOW_WHITE

rem note the -format %c is -format %%c - ie %% is escaping for literal %

SET exitcode=
rem the for next loop IN (a,s,t) reads from a step s to t
FOR /L %%a IN (63,1,71)  DO (
     convert %%a.jpg -format %%c -depth 1 histogram:info:- > histogramx8.tmp
     call :eightlinestoone "%%a"
                            )

rem post processing house keeping

del histogramx8.tmp
exit /B


rem sub routines being called from main

```

```
:outresult
rem output is tab deliminated with fieldname & data-value pairs
rem tilde ~ removes double quote on output
echo CMD:histogramx8    FILE:%~1.jpg!%2!

exit /B


:eightlinestoone
set "textline="
for /f "tokens=*" %%b in (histogramx8.tmp) do (
  set "textline=!textline!    %%b"
)
call :outresult %1 %textline

exit /B


rem the following line is a sample output of script
rem CMD:histogramx8 FILE:71.jpg     88557: (  0,  0,  0) #000000 black      535: (  0,  0,255) #0000FF blue 3617: (  0,255,  0) #00FF00 lime        315: (  0,255,255) #00FFFF cyan 9674: (255,  0,  0) #FF0000 red 214: (255,  0,255) #FF00FF magenta      5552: (255,255,  0) #FFFF00 yellow      11536: (255,255,255) #FFFFFF white

rem we do not need all!?

rem what we really need is the following (ie need to parse string on input)
CMD:histogramx8 FILE:71.jpg     88557:black      535:blue 3617:lime        315:cyan 9674:red 214:magenta      5552:yellow 11536:white
```

Script: tesselate.bat

Break an image (landscape) into tiles/parts

```
echo off
setlocal enableextensions enabledelayedexpansion

rem window 10 script in cmd
rem jv dec 2106

rem depends on imagemagick installed (with legacy option if v >= 7)
rem depends on Corel 1000 low res images and named as 1.jpg, 2.jpg, ..., 1000.jpg
rem assumes presence of sub-dirs to save tesselations (one per size)

rem invoke scripts\tesselate.bat
rem invoke scripts\tesselate.bat > whatever.lst

rem works only for landscape images

rem convert 1.jpg -format "%[fx:(w/h>1)?1:0]" info:
rem convert -quality 100 -crop 100x100 1.jpg t12_100wby100h/1_tile100by100_%02d.jpg

rem header line
echo CMD    FILE

rem note the -format "% is -format "%% - ie %% is escaping for literal %
rem note same for destination filename

SET exitcode=
rem the for next loop IN (a,s,t) reads from a step s to t
FOR /L %%a IN (63,1,71)  DO (
     convert %%a.jpg -format "%%[fx:(w/h>1)?1:0]" info: > exitcode.tmp
     set /P exitcode=<exitcode.tmp
     IF !exitcode! EQU 1 (
         convert -quality 100 -crop 100x100 %%a.jpg t12_100wby100h/%%a_tile100by100_%%02d.jpg
         convert -quality 100 -crop 200x150 %%a.jpg t4_200wby150h/%%a_tile200by105_%%02d.jpg
         convert -quality 100 -crop 200x300 %%a.jpg t2_200wby300h/%%a_tile200by300_%%02d.jpg
         call :outresult "%%a"
                         )
                           )

rem post processing house keeping

del exitcode.tmp
exit /B


rem sub routines being called from main

```

```
:outresult
rem output is tab deliminated with fieldname & data-value pairs
echo CMD:tesselate_landscape_12_4_2 FILE:%1.jpg
exit /B
```

Script: cmpjpg.bat

Compare two images

```
@echo off

rem window 10 script in cmd
rem jv dec 2106

rem depends on imagemagick installed (with legacy option if v >= 7)
rem depends on Corel 1000 low res images and named as 1.jpg, 2.jpg, ..., 1000.jpg

rem invoke scripts\cmpjpg.bat
rem invoke scripts\cmpjpg.bat > whatever.lst

rem 'convert -fuzz 10%    -> how much tolerance for an colour variation to accept as the same (use in jpegs)
rem 'compare -metric ae   -> absolute error at pixel level and returns the different ones (out of length x width in pixels) (o = equal )
rem 'compare -metric ncc  -> normalized cross correlation (1 = similar)

rem header line
echo CMD   FILE1 FILE2 AE    NCC

rem note the -fuzz 10% is -fuzz 10%% - ie %% is escaping for literal %
rem the for next loop IN (a,s,t) reads from a step s to t

FOR /L %%a IN (1,1,5)  DO (
     FOR /L %%b IN (1,1,5) DO (compare -fuzz 10%% -metric ae  %%a.jpg %%b.jpg -compare 2>t1.tmp
                              compare -fuzz 10%% -metric ncc %%a.jpg %%b.jpg -compare 2>t2.tmp
                              call :outresult "%%a" "%%b"
                             )
                          )

rem post processing house keeping
del t1.tmp
del t2.tmp
exit /B

rem sub routines being called from main
:outresult
set /P ae=<t1.tmp
set /P ncc=<t2.tmp
rem output is tab deliminated with fieldname & data-value pairs
echo CMD:compare_fuzz10pc  FILE1:%1.jpg    FILE2:%2.jpg     AE:%ae%    NCC:%ncc%
exit /B
```

Database stuff

Create Database, Schemas (and some tables)

```
-- create db
CREATE DATABASE cbir_corel10k
  WITH ENCODING='UTF8'
       OWNER=postgres
       CONNECTION LIMIT=-1;
```

```
-- create  schema
CREATE SCHEMA cbir
       AUTHORIZATION postgres;
-- create  schema
-- hold generic routines that are not subject related
CREATE SCHEMA dev
       AUTHORIZATION postgres;
 -- Use schema public as temp
```

```
-- test data import from O/S
-- use when one logical record is depicted in one physical line
drop table if exists sample;
create table sample
(i    serial primary key,
 call_i     integer,
 txt  character varying (999),
 ltxt text);

-- test data import from O/S
-- use when one logical record is depicted in many physical lines
drop table if exists manyline;
create table manyline
(i    serial primary key,
 call_i     integer,
 txt  character varying (999),
 ltxt text);
```

-- win 7,8, 10 issues with rights to import

-- create a temp folder on c:\ called temp

-- check who is the postgresql service owner (e.g. NETWORK SERVICE)

-- go to temp folder properties and give m r&e, l, r, and w rights (ALLOW) to NETWORK SERVICE

-- test with the following data saved in some file (eg. colourval.lst)

```
    113316: (  0,  0,  0) #000000 black
        13: (  0,  0,255) #0000FF blue
         7: (  0,255,  0) #00FF00 lime
        25: (  0,255,255) #00FFFF cyan
      5563: (255,  0,  0) #FF0000 red
         5: (255,  0,255) #FF00FF magenta
       821: (255,255,  0) #FFFF00 yellow
       250: (255,255,255) #FFFFFF white
```

```
-- read an ascii text file (convert the above and save it in c:\temp)
-- (note pk is generated by server)
 copy sample(txt)
 from 'c:\\temp\\colourval.lst';
 -- or (need to find, compile stored procedure – elsewhere in text)
 select dev.execCopyIn('sample(txt)', 'c:\\temp\\colourval.lst');
 -- check outcome - should be the above with PK set

-- read from output of a program (note pk is generated)
-- eg type filename
copy sample(txt)
from program 'type c:\\temp\\colourval.lst';
 -- check outcome - should be (another of) the above with PK set

-- read from output of a program (note pk is generated)
copy sample(txt)
from program 'convert C:\\Users\\...\\cbirCorel10k\\1.jpg -format "%[fx:(w/h>1)?1:0]" info:'
```

General stored procedures

```
create or replace function dev.execCopyIn(intableexp text, infullfname text) returns void as
$body$
declare
  txt_cmd character varying(299);
begin
  SET client_encoding = 'WIN1258';
  -- does not accomodate delimeter and header etc
  txt_cmd := format('copy %s from ''%s'' encoding ''WIN1258''', intableexp, infullfname);
  -- raise notice 'execCopyIn cmd is %', txt_cmd;
  execute txt_cmd;
end
$body$ language plpgsql;
```

```
create or replace function dev.execCopyProgIn(intableexp text, infullfname text) returns void as
$body$
declare
  txt_cmd character varying(299);
```

```
begin
  SET client_encoding = 'WIN1258';
  -- does not accomodate delimeter and header etc
  txt_cmd := format('copy %s from program ''%s'' encoding
''WIN1258''', intableexp, infullfname);
  -- raise notice 'execCopyProgIn cmd is %', txt_cmd;
  execute txt_cmd;
end
$body$ language plpgsql;
```

Rem check a client's session (e.g. pgAdminIII) setting, set setting, and reset

SHOW client_encoding;

SET client_encoding = 'WIN1258';

RESET client_encoding;

```
create or replace function dev.whatserverencoding() returns
character as
$body$
declare encode_str character(99);
begin
    SELECT pg_encoding_to_char(encoding)::character(99)
    into encode_str
    FROM pg_database
    WHERE datname = 'cbir_corel10k';
   return encode_str;
end
$body$ language plpgsql;

select dev.whatserverencoding();
```

Basic ETL scripts to load details of images dataset

```
drop table if exists cbir.srgb;

create table cbir.srgb
(sc_id            character varying (25) primary key,
 sc_hex     character varying (25) not null,
 sc_red     integer not null,
 sc_green   integer not null,
 sc_blue    integer not null
 );
 insert into cbir.srgb(sc_id,sc_hex,sc_red,sc_green,sc_blue)
values
 ('black',  '#000000',  0,  0,  0),
 ('blue',   '#0000FF',  0,  0,255),
 ('lime',   '#00FF00',  0,255,  0),
 ('cyan',   '#00FFFF',  0,255,255),
 ('red',    '#FF0000',255,  0,  0),
 ('magenta','#FF00FF',255,  0,255),
 ('yellow', '#FFFF00',255,255,  0),
 ('white',  '#FFFFFF',255,255,255);


drop table if exists cbir.dataset;

create table cbir.dataset
(ds_id      serial primary key,
 ds_handle character varying(999) unique,
 ds_filename character varying(999),
 ds_filetype character varying(99),
 ds_height smallint,
 ds_width   smallint,
 ds_hash    character varying(99),
 ds_colourspace character varying(99),
 ds_original character varying(9) default 'YES',
 ds_details text);


-- etl into dataset

 -- TWO MAIN ISSUES (re coding)

 -- RIGHTS
 -- win 7,8, 10 issues with rights to import
 -- check who is the postgresql service owner (e.g. NETWORK SERVICE)
 -- go to jpg dataset folder properties and give fc, m r&e, l, r, and w rights (ALLOW) to NETWORK SERVICE

 -- CHARACTER ENCODING OF DATA (windows file?), CLIENT SESSION (set this!), SERVER (ie UTF8)
 SHOW client_encoding;
 SET client_encoding = 'WIN1258';
 -- RESET client_encoding;


-- LOADING into dataset

-- load into table sample
copy sample(txt)
from program 'C:\\Users\\...\\cbirCorel10k\\scripts\\uploadall.bat';
-- Query returned successfully: 1000 rows affected, 45.1 secs execution time.

insert into cbir.dataset
    (ds_handle, ds_filename, ds_filetype,
     ds_height, ds_width,    ds_hash,
     ds_colourspace)
select
   trim(trim((txt::json->'filename')::text,'"'),'.jpg'),
   trim((txt::json->'filename')::text,'"'),
   trim((txt::json->'filetype')::text,  '"'),
   trim((txt::json->'height')::text,  '"')::smallint,
   trim((txt::json->'width')::text,  '"')::smallint,
   trim((txt::json->'hash')::text,  '"'),
   trim((txt::json->'colourspace')::text,  '"')
from sample;

delete from sample;



create or replace function cbir.upd_dataset_details() returns void as
$body$
declare
  txt_verbose character varying(99);
  txt_cmd character varying(299);
begin
  SET client_encoding = 'WIN1258';
  -- open cursor per file
  delete from sample;
  delete from manyline;
  insert into sample(txt)
    select ds_filename from cbir.dataset where ds_details is null limit 100;  --ds_filename='604.jpg';

  -- copy with identify verbose into temp (ltxt)
  for txt_verbose in select txt from sample loop
     raise notice 'current image %', txt_verbose;
     txt_cmd := format('identify -verbose C:\\Users\\...\\cbirCorel10k\\%s',txt_verbose);
     perform dev.execCopyProgIn('manyline(txt)', txt_cmd);
     -- convert many lines into one line
     -- move into table
     update cbir.dataset
       set ds_details = (select array_to_string(array(select convert_to(txt,'UTF8') from manyline order by i), chr(13))::text)
       where ds_filename=txt_verbose;
     delete from manyline;
  end loop;

end
$body$ language plpgsql;

select cbir.upd_dataset_details();  -- run this multiple times (note the limit 100 clause)
```