-- We don't want the tags. How do we get rid of them?

-- There is a function text() that goes INTO the XPath!!
-- NOT after it. text() works only for ONE LEVEL.

select Name, a.card.extract('/acard/email/text()').getstringval()
from addr_book a
where a.card.existsNode('/acard') = 1

-- James and Roger have two phone numbers.
-- By default both are returned. Without Blank in between!

select Name, a.card.extract('/acard/wphone/text()').getstringval()
from addr_book a
where name='James' or name='Roger'

-- Not good...

-- What if we want both phone numbers?

select Name, a.card.extract('/acard/wphone[1]/text()').getstringval(),
a.card.extract('/acard/wphone[2]/text()').getstringval()
from addr_book a
where name='James' or name='Roger'
/

-- If out of bound, no error. Just nothing returned.

select Name, a.card.extract('/acard/wphone[1]/text()').getstringval(),
a.card.extract('/acard/wphone[2]/text()').getstringval(),
a.card.extract('/acard/wphone[3]/text()').getstringval()
from addr_book a
where name='James' or name='Roger'
/

-- What if we want both phone numbers nicely separated?

select Name,
a.card.extract('/acard/wphone[1]/text()').getstringval() || ' ** ' ||
a.card.extract('/acard/wphone[2]/text()').getstringval()
from addr_book a
where Name = 'Roger';

p48

-- A longer path:

select Name, a.card.extract('/acard/address/zip/text()').getstringval()
from addr_book a
where Name = 'Bobby'

p49

--  * matches one element.
-- This stands for ONE LEVEL.
-- Note that below I skip the "address"

select Name, a.card.extract('/acard/*/zip/text()').getstringval()
from addr_book a
where Name = 'Bobby'
/

-- This does not give the zip code. Can't skip two levels.

select Name, a.card.extract('/*/zip/text()').getstringval()
from addr_book a
where Name = 'Bobby'
/

This DOES work.

select Name, a.card.extract('/*//*/zip/text()').getstringval()
from addr_book a
where Name = 'Bobby'
/

p50

-- You can use wildcards in paths. // matches all descendants.
-- That means.... everything under the //
-- Note that below I skip the "address"

select Name, a.card.extract('/acard//zip/text()').getstringval()
from addr_book a
where Name = 'Bobby'
/

-- Or even this:
select Name, a.card.extract('//zip/text()').getstringval()
from addr_book a
where Name = 'Bobby'

-- Why? Because I don't want to remember where in the tree
-- zip is stored. But notice: Search time goes up!

select Name, a.card.extract('//wphone/text()').getstringval()
from addr_book a
where Name = 'Roger'

-- You can use something like an implicit WHERE statement in the path.

select Name,
a.card.extract('/acard[email=roger12@yahoo.com]//zip/text()').getstringval()
from addr_book a

-- The above gets the ZIP code. But... only for an acard that
-- has under it an email as given.
-- Note: The above only works with DOUBLE QUOTES.
-- Also note, as usual it returns all the rows anyway.
-- The extraction does not affect the where clause.

select Name,
a.card.extract('/acard[email=roger12@yahoo.com]//zip/text()').getstringval()
from addr_book a
where
a.card.existsNode('/acard[email=roger12@yahoo.com]//zip/text()') = 1

-- You can even have a path inside of the [ ] !!!!
This below says: Give me the Name and email address of everybody who lives
in Atlanta, but show me all the names of people with email addresses.

select Name,
a.card.extract('/acard[address/city="Atlanta"]//email/text()').getstringval()
from addr_book a
where
a.card.existsNode('/acard/email') = 1
/

-- You can extract in a where clause also.
select Name, a.card.extract('/acard/email/text()').getstringval()
from addr_book a
where a.card.extract('/acard/address/city/text()').getstringval() = 'Atlana'

-- It's wrong in the database, so I need it wrong here too!!!!
-- Give me the email address of people who live in Atlanta.
-- If you want a string, you have to now append the method
-- getstringval()
-- If you want a number, you use .getnumberval()
-- This is almost the end of processing XML within Oracle.
-- Of course there is much more to learn about XPath...
-- REMEMBER XML PRESERVES WHITESPACE.

THUS
<num>234</num>
IS NOT THE SAME AS
<num> 234 </num>
AND
<num>
234
</num>

-- You can extract attributes also, with an @ sign:

select Name, a.card.extract('/acard/email/text()').getstringval()
from addr_book a
where a.card.extract('/acard/@createdby').getstringval() = 'Jim'
/

select Name, a.card.extract('/acard/@createdby').getstringval()
from addr_book a
/

p52c
select * from cat
select * from tab
select * from user_catalog
select * from user_objects
select * from user_source       -- actual code of procedures
select * from user_tables
select * from all_tables
select * from user_tab_columns
select * from user_views
select * from user_triggers
select * from user_procedures   -- list of procedures
select * from user_tab_privs    -- permissions
select * from user_errors       -- undisplayed error messages
select * from user_triggers

p53

-- Brief review of another basic topic:  System tables.
-- See also the file systemtables.

-- select * from cat;

-- describe cat

-- select table_name from cat;
-- But this is still "single column"

-- So, now I can use what I learned in PL/SQL for a nicer display.

create or replace procedure ls
as
  i  number := 0;

begin
  for anothercursor in
    (select table_name from cat)
  loop
    dbms_output.put_line(anothercursor.table_name);
  end loop;
end;
/

begin
  ls;
end;
/