**Notes**

Pg 16

1.) select Sname, Major

from schClasses sc, Enrollments e, students s

Where year=2013 and sem='Sp'

And Dept='IS' and Cnum=300

And sc.CallNum=e.CallNum

And e.snum=s.snum;

2.) And major!='IS'

3.) select dept, Cnum, ctitle

from courses c, enrollments e, schclasses sc

Where s.snum=101 and sc.semester='Sp'

And s.snum = e.snum

And sc.dept = c.dept

And sc.cnum = c.cnum;

Ex 2.2

1.) select count(Snum)

From students,

Where major='IS';

2.)

Select count (Snum)

From students;

7.) select sum(crHr)

From enrollments e, students s, schClasses sc

Where grade != 'f'

And e.Sum = 101

And e.callnum = sc.callnum

And sc.dept c.Dept and sc.cum = c.cnum

HW #1: pg 16 ex2.1 #5, pg 18 ex2.4 #8, pg 16, 8 ex2.5 #, pg 26 ex4.1 #1, #3

Due next Wednesday

Need to add customers and amazon tables

--2.2 #9

select sum(grPt\*CrHr) / sum(crHr)

from enrollments e, schclasses sc, students s

where e.snum = 101

and e.callnum = sc.callnumand sc.dept = c.dept

and sc.cnum=c.cnumand grade !='F';

--2.3 #1

select sum(amount)

from amazon.com

where custNum = 101;

--2.3 #2

select sum(amount)/(2019 - (min(to\_char(OrdDate,'yyyy')))) from amazon where custNum = 101;

create table amazon(custNum number(3),OrdDate date,amount number(5));

--2.4 #4

select avg(GPA), major

from students

group by major;

--2.4 #5

select major

from students

group by major

having count(SNum) > 100;

--2.4 #6

select major

from students

group by major

having avg(gpa) > 3.5;

--2.4 #7

select major

from students

where status='Probation'

group by major

having count(snum) > 35;

--2.5 #1

select count(custNum)

from customers

group by gender;

--2.5 #2

select count(custNum)

from customers

group by Prime;

--2.5 #3

select gender, sum(amount)

from amazon.com a, customers c

where a.custNum, c.custNum

group by gender;

--2.5 #4

select prime, sum(amount)

from amazon.com a, customers c

where a.custNum, c.custNum

group by prime;

--2.5 #5

select gender, sum(amount)/(count(customers.custNum))

from amazon.com a, customers c

where a.custNum = amazon.com.custNum

group by gender;

--2.5 #6

select prime, sum(amount)/(count(customers.custNum))

from amazon.com a, customers c

where a.custNum = amazon.com.custNum

group by prime;

--character dunctions

--Concat:

--Select Last || ',' || First from students;

--for header to look better

--Select Last || ',' || First as sname from students;

--SNAME

--Smith, John

upper('sOphIe') -- -> SOPHIE

lower('') --> all lower

initcap('') --> Sophie

initcap('soPHIE lEE') --> Sophie Lee

rtrim('Sophie ') --> 'Sophie'

ltrim(' Sophie') --> 'Sophie'

ltrim(rtrim(' Sophie ')) --> 'Sophie' (does both sides at once)

--trim whenever user input. Don;t trust users!

ltrim(rtrim(' ')) --> null

rtrim('Sophie\*\*\*\*\*', '\*') --> Sophie

select rpad(pnum, 8,' ') --> 'p1 ' (8 characters wide)

--rpad helps control width of the columns

--substrings

substr('sopHIe',4,2) --> HI

substr('sOPHIE', 2) --> OPHIE

substr('sophIE', -2) --> IE (last two)

instr('xyz-8251', '-') --> 4 (gives index of where it is)

instr('xyz-8251', 'A') --> 0 (index not found)

--only finds first instance of char. run loop to find multiple

length('1234567ABC') --> 10 (it returns how long the string is)

--4.8 #1

update students

set username= 'z' || substr(lastname, 1, 2)

|| substr(SSN, -4);

--4.9 #5

--update and concat prefix and suffix of hyphen

ceil(3.45) --> 4

floor(3.45) --> 3

round(123.45) --> 123

round(123.45, 1) -->123.5

round(123.45, -1) --> 120 (rounds to one place left of the decimal point)

trunc() --> takes off the digits after the decimal

mod(9,4) -->1

select decode(grade, 'A', 4, 'B', 3, 'C', 2, 'D', 1, 'F', 0, 0) --> last digit is value for else (if no match)

from enrollments;

select year semester

from e order by decode(semesester, 'sp',1,'su',2,'fa',3,0);

select Tr#, TrDate

decode(trType,'Credit', Amount, -Amount)

from ....;

--decode as null, x, then can count. will not count the null

--group by M or F

nvl(major,'NA') != 'IS' --if major is null, the use 'NA' as value instead--Topic 4: union, intersect, minus

--union all will not eliminate duplicates

--must be union compatable to do union

--must have same structure (like number of columns)

--must have same data type in corresponding columns

--4.1 #1

(select students.SNum

from students)

minus

(select enrollments.SNum

from enrollments

where courseNum = 'MGT 425');

--4.1 #2

(select students.SNum

from students

where major='IS');

minus

(select enrollments.SNum

from enrollments, students

where courseNum = 'IS 380' and major = 'IS'

and enrollments.SNum = students.SNum);

--4.1 #3

(select SNum

from enrollments

where courseNum = 'IS 380')

intersect

(select SNum

from enrollments

where courseNum = 'IS 385');

--4.1 #4

(select SNum

from enrollments

where courseNum = 'IS 380')

minus

(select SNum

from enrollments

where courseNum = 'IS 300')

--4.1 #5

(select SNum

from enrollments)

minus

(select SNum

from enrollments

where semester = 'spring')

--self join is a way to join a table twice

--in case of courses join with prereq to get the titles for each

--can't join two different cNum to one cnum in other table

select p.cnum, c1.ctitle, p.precnum, c2.ctitle

from prereq p, courses c1, courses c2

where p.cnum = c1.cnum and p.precnum = c2.cnum;

--selecting the title from the table you joined together

--outer join

--if join column has null values, when go to join it will get rid of records because no matching value

select

from students s, majors m

where s.major = m.major(+);

SNUM | SNAME | MAJOR-(has nulls)

-----------------------------------

select

from students s, majors m

where s.major(+) = m.major;

SNUM-(has nulls) | SNAME-(has nulls) | MAJOR

----------------------------------------------

--put (+) on the side where you allow null values

--6.1 q 4

select \*

from orders

where amount > (select avg(amount) from orders);

SELECT prod\_ord.\*, productDescription.category, productDescription.prodName, prod\_ord\_Query.price

FROM productDescription INNER JOIN prod\_ord ON

productDescription.prodID = prod\_ord.prodNum;

select \*

from orders

where amount = (select max(amount)

from orders);

delete from orders

where pnum in (select pnum from products where status='Disct');

--8

update customers

set member='Gold'

where cnum in (select cnum

from orders

group by

having sum(amount) > 10000);

--comparing 'where' cnum to 'select' cnum in subquery

--creating new table from existing

create table students2 as (select \* from students);

--for only certain columns:

create table students2 as (select snum, sname, major

where standing = 'Senior');

--add in more records to existing table

insert into students2

select \*

from students;

--joining subquery with another query to get same level (multi & multi)

select sc.callnum, sc.cap, temp1.currEnrolled

from

(select callnum, count(snum) currEnrolled

from enrollments

group by callnum) temp1,

schClasses sc

where temp1.callnum = schclasses.callnum;