Relational Algebra/SQL Practice problems – Set 2 (86 points)

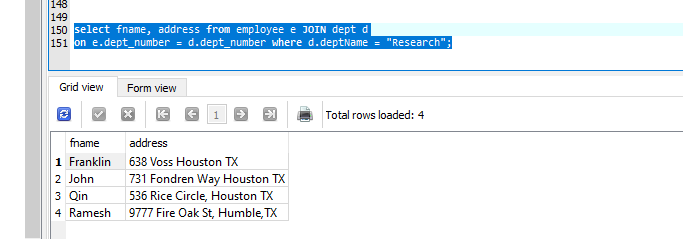
Schemas and instance are provide on the ‘PS2\_tables and Data’ spread sheet. [PS2\_tables and Data.xlsx](https://uncc.instructure.com/courses/146088/files/11967895?wrap=1)[download](https://uncc.instructure.com/courses/146088/files/11967895/download?download_frd=1)

1. Write the relational query, the SQL and then show the results of the following queries given the design shown in tab 5.5 and results in the instance of the schema in tab 5.6 in the spreadsheet. You may use an aggregate function as needed.
   1. The full name and address of all employees who work for the Research department.

ρEMP🡨 Employee  EMPLOYEE.dept\_number = DEPARTMENT.dept\_number Dept

Result 🡨Πfname, address(σdeptName = "Research") (EMP)

select fname, address from employee e JOIN dept d on e.dept\_number = d.dept\_number where d.deptName = "Research";



* 1. For every project located in the town Stafford, list the project number, the controlling department number and the department manger’s last name, address and date of birth.

Duplicating the JOIN for Department to have two comparison(department number and employee id)

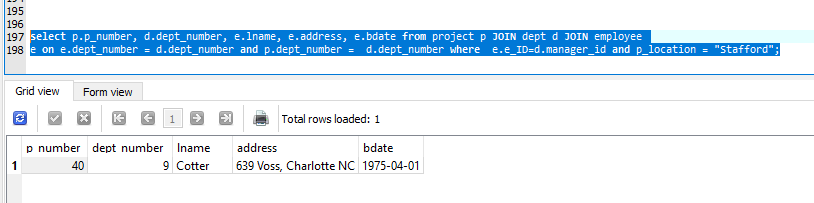
ρPROJ🡨 Project  PROJECT.dept\_number = DEPARTMENT.dept\_number Dept

ρEMP 🡨 Employee EMPLOYEE.dept\_number = PROJ.dept\_number  PROJ

ρEMPDEP 🡨 Dept EMPLOYEE.e\_id = PROJ.manger\_id EMP

Result 🡨Πp\_number, dept\_number, lname, address, bdate(σp\_location="Stafford") (EMPDEP)

select p.p\_number, d.dept\_number, e.lname, e.address, e.bdate from project p JOIN dept d JOIN employee e on e.dept\_number = d.dept\_number and p.dept\_number = d.dept\_number where e.e\_ID=d.manager\_id and p\_location = "Stafford";



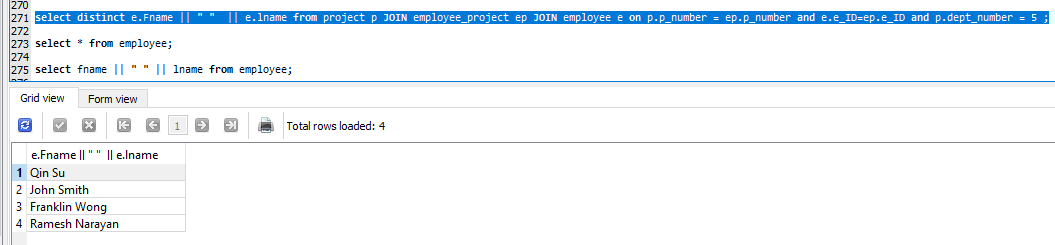
* 1. Find the full names of employees who work on all of the prject controlled by department 5.

ρPROJ🡨 Project  PROJECT.p\_number = EMPLOYEE\_PROJECT.p\_number Emploee\_Project

ρEMP 🡨 Employee EMPLOYEE.e\_ID = PROJ. e\_ID  PROJ

Result 🡨Π(fname) append (lname) (σ dept\_number = 5) (EMP)

select distinct e.Fname || " " || e.lname from project p JOIN employee\_project ep JOIN employee e on p.p\_number = ep.p\_number and e.e\_ID=ep.e\_ID and p.dept\_number = 5 ;

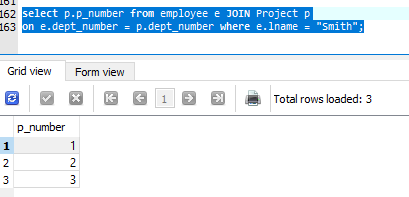


* 1. Provide a list of all project numbers that have an employee with a last name of Smith working on them in any role.

ρEMPL🡨 Employee  EMPLOYEE.dept\_number = PROJECT.dept\_number Project

Result 🡨Πp\_number(σlname = "Smith")(EMPL)

select p.p\_number from employee e JOIN Project p on e.dept\_number = p.dept\_number where e.lname = "Smith";



5.List the name(s) of all employees have have two or more dependents.

ρEMPL🡨 Employee  EMPLOYEE.e\_ID = DEPRNDENT.e\_id Dependent

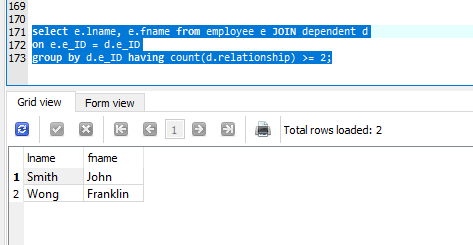
Result 🡨 Πlname,fname (γe\_id σ count(relationship) >= 2) (EMPL)

γ – is for Group by

Highlighted expression is for getting the value from GROUP BY HAVING

select e.lname, e.fname from employee e JOIN dependent d

on e.e\_ID = d.e\_ID group by d.e\_ID having count(d.relationship) >= 2;



6.List the name(s) of all employees who have no dependents

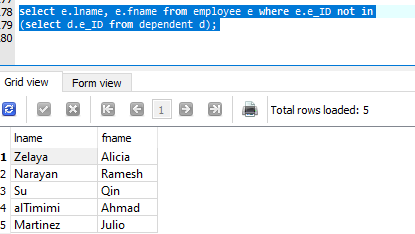
ρEPEID 🡨 σ e\_id(DEPENDENT)

Result 🡨 Πlname,fname (σ e\_id != EPEID ) (EMPLOYEE)

Highlighted expression is for sub query

select e.lname, e.fname from employee e where e.e\_ID not in

(select d.e\_ID from dependent d);



7.List the name(s) of all employees who have at least one dependent and whose role is Manager in at least one project.

Note: There is no **Role** column so I am using manager id.

ρEMPL🡨 Employee  EMPLOYEE.e\_ID = DEPRNDENT.e\_id Dependent

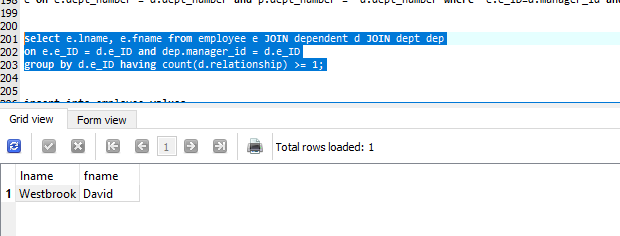
ρEMPDEP🡨 Dept  EMPL.e\_ID = DEPT.manager\_id EMPL

Result 🡨 Πlname,fname (γe\_id σ count(relationship) >= 1) (EMPDEP)

select e.lname, e.fname from employee e JOIN dependent d JOIN dept dep

on e.e\_ID = d.e\_ID and dep.manager\_id = d.e\_ID

group by d.e\_ID having count(d.relationship) >= 1;



2.Specify the following queries on the database schema shown in tab 5.5 and show the results of each query as they would apply to the instance shown in tab 5.6

* 1. Retrieve the names of all employees in department 5 who work more than 10 horus per week on the SeqN project.

ρPROJ🡨 Project  PROJECT.p\_number = EMPLOYEE\_PROJECT.p\_number Employee\_Project

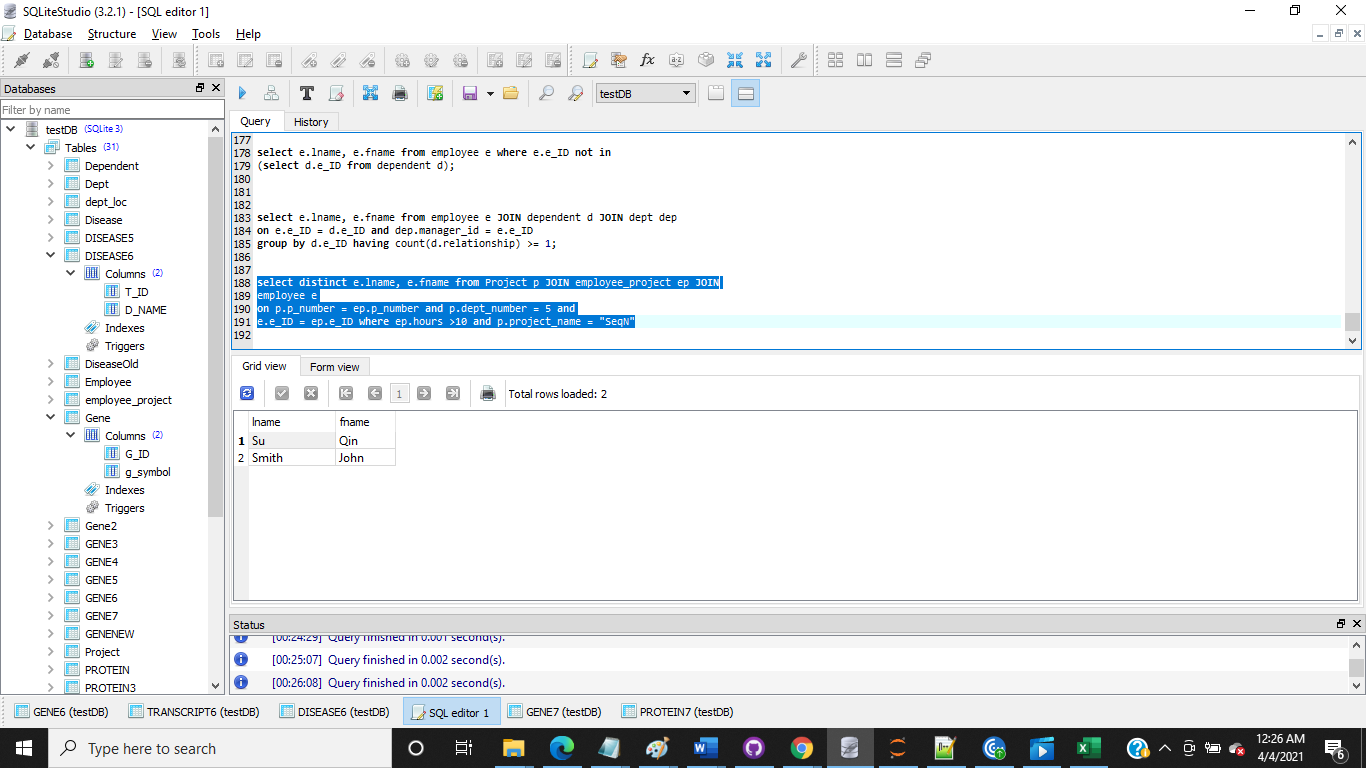
ρEMP 🡨 Employee EMPLOYEE.e\_ID = FLIG. e\_ID  PROJ

Result 🡨Π lname, fname (σ (dept\_number = 5) and (hours >10) and (project\_name = "SeqN")) (EMP)

select distinct e.lname, e.fname from Project p JOIN employee\_project ep JOIN

employee e on p.p\_number = ep.p\_number and p.dept\_number = 5 and

e.e\_ID = ep.e\_ID where ep.hours >10 and p.project\_name = "SeqN"



* 1. List the names of all employees who have a dependent with the same first name as themselves.

Using SELF JOIN on EMPLOYEE table

ρEMPL🡨 Employee  EMPLOYEE.e\_ID = DEPENDENT.e\_ID Dependent

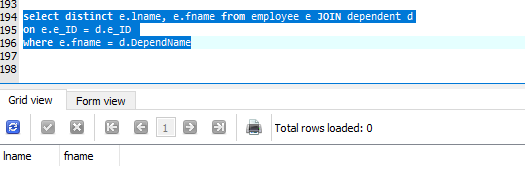
ρEMPDEP 🡨 Employee EMPLOYEE.fname = EMPL.dependName  EMPL

Result 🡨Πlname, fname(EMPDEP)

select distinct e.lname, e.fname from employee e JOIN dependent d

on e.e\_ID = d.e\_ID

where e.fname = d.DependName



* 1. Find the names of all employees who are supervised by Franklin Wong.

Way 1:

EMPID 🡨Π e\_ID(σ (Fname ="Franklin") and (lname = "Wong"))(EMPLOYEE)

Result 🡨Πlname, fname(σsupervisor\_id=empid) (EMPLOYEE)

Highlighted expression is for sub query

Way 2:

Using SELF JOIN:

EMPID 🡨Π e\_ID(σ (Fname ="Franklin") and (lname = "Wong"))(EMPLOYEE)

ρEMPL🡨 Employee  EMPLOYEE.e\_ID = EMPLOYEE.e\_ID Employee

Result 🡨Πlname, fname(σsupervisor\_id=empid) (EMPL)

select e.lname, e.fname from employee e JOIN employee e1

on e.e\_ID = e1.e\_ID where e.supervisor\_id

in( select e\_ID from employee where Fname ="Franklin" and lname = "Wong");

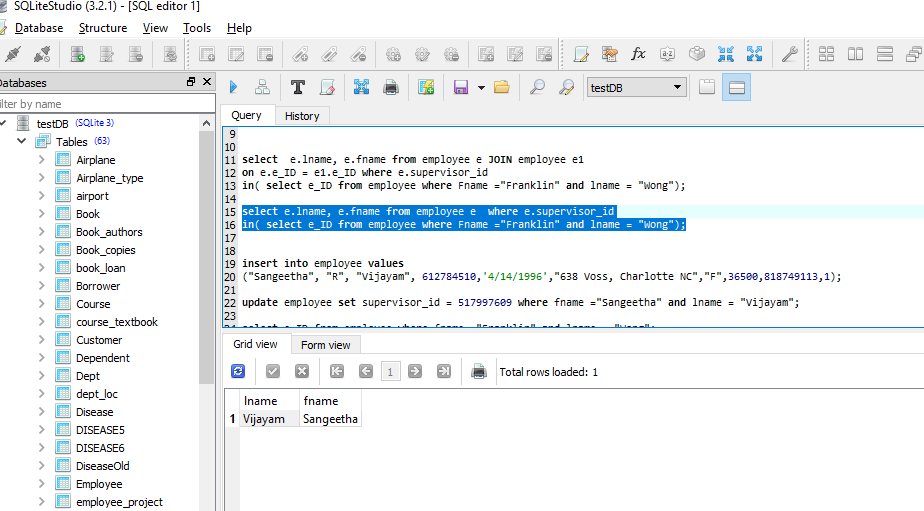
Using Sub Query:

select e.lname, e.fname from employee e where e.supervisor\_id

in( select e\_ID from employee where Fname ="Franklin" and lname = "Wong");

Note: Added a record for getting valid result

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Fname** | **initial** | **lname** | **e\_ID** | **……………………………….** | **supervisor\_id** | **dept\_number** |
| Sangeetha | R | Vijayam | 612784510 | ……………………………….. | 517997609 | 1 |



* 1. For each project, list the project name and the total hours per week of all employees spent on the project, list the projects in descending order of total hours spent per week.

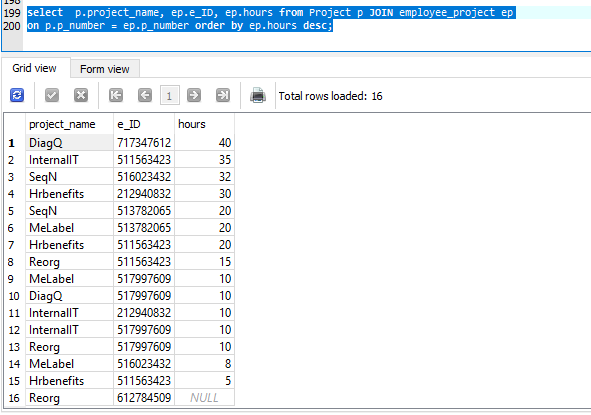
ρPROJ🡨 Project  PROJECT.p\_number = EMPLOYEE\_PROJECT.p\_number Emploee\_Project

Result 🡨 REVERSE(**T** hours ( Π project\_name, e\_ID, hours(PROJ) ))

**T -**(tau) operator for order by

select p.project\_name, ep.e\_ID, ep.hours from Project p JOIN employee\_project ep

on p.p\_number = ep.p\_number order by ep.hours desc;



* 1. Retrieve the names of all employees who work on SeqN, MeLabel and the DiagQ projects.

ρEMP🡨Employee EMPLOYEE.p\_number=EMPLOYEE\_PROJECT.p\_number Employee\_Project

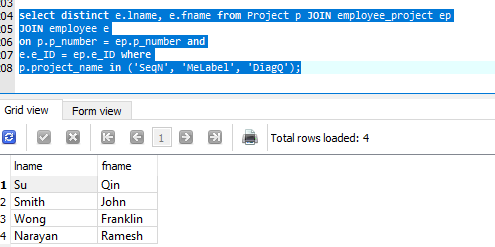
ρPROJ🡨 Project  PROJECT.e\_id = EMPLOYEE\_e\_id EMP

Result 🡨Πlname, fname(σ(project\_name =”SeqN”) or (project\_name = “MeLabel”) or (project\_name = “DiagQ”)) (PROJ)

select distinct e.lname, e.fname from Project p JOIN employee\_project ep

JOIN employee e on p.p\_number = ep.p\_number and

e.e\_ID = ep.e\_ID where p.project\_name in ('SeqN', 'MeLabel', 'DiagQ');



* 1. Retrieve the names of any employees who work only on the InternalIT, Reorg and Hrbenefits projects.

ρEMP🡨Employee EMPLOYEE.e\_id=EMPLOYEE\_PROJECT.e\_id Employee\_Project

ρPROJ🡨 Project  PROJECT.dept\_number = EMP.dept\_number EMP

Result🡨Πlname, fname(σ(project\_name =”SeqN”) or (project\_name = “MeLabel”) or (project\_name = “DiagQ”)) (PROJ)

Way 1:

select distinct e.Fname from employee e JOIN project p JOIN employee\_project ep

on e.e\_id = ep.e\_id and e.dept\_number = p.dept\_number and p.project\_name in ("InternalIT", "Reorg","Hrbenefits");

* 1. For each department, retrieve the department name and the average salary of all employees working in the department.

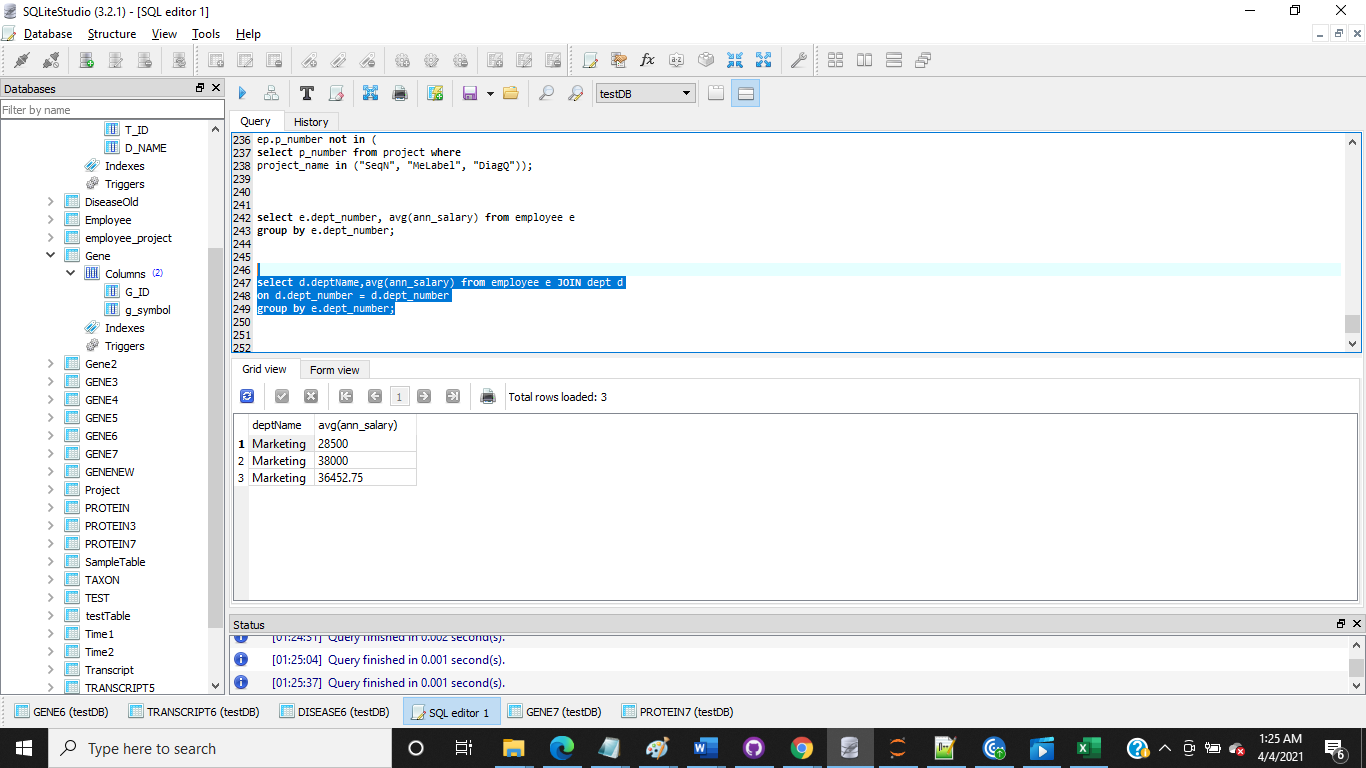
ρEMPL🡨 Employee  EMPLOYEE.dept\_number = DEPARTMENT.dept\_number Dept

Result 🡨ΠdeptName,average(ann\_salary) (γdept\_number) (EMPL)

select d.deptName,avg(ann\_salary) from employee e JOIN dept d

on e.dept\_number = d.dept\_number

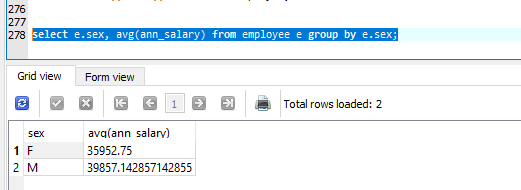
group by e.dept\_number;



* 1. Retrieve the average salary of all female employees and of all male employees.

Πsex, average(ann\_salary) (γsex)(Employee)

select e.sex, avg(ann\_salary) from employee e group by e.sex;



* 1. List the last names of all department managers who have no dependents.

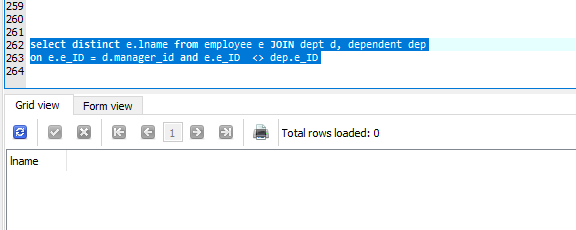
ρEMPL🡨 Employee  EMPLOYEE.e\_ID = DEPARTMENT.manager\_id Dept

ρEMPDEP🡨 Dependent  EMPL.e\_ID != DEPENDENT.e\_ID EMPL

Result 🡨Πlname (EMPDEP)

select distinct e.lname from employee e JOIN dept d, dependent dep

on e.e\_ID = d.manager\_id and e.e\_ID <> dep.e\_ID



* 1. List the name of all employees who work on a project whose location is different from the city given in their home address.

Note: As the question didn’t ask for same department I didn’t add that in the query

Πlname, fname(σaddress not like p\_location)Employee  Project

select distinct Fname, lname from employee e JOIN project p WHERE address NOT LIKE '%' || p.p\_location || '%';

ρEMPL🡨 Employee  EMPLOYEE.dept\_number = PROJECT.dept\_number Project

Result 🡨Πlname, fname(σaddress not like p\_location)(EMPL)

select distinct Fname, lname from employee e JOIN project p on e.dept\_number = p.dept\_number WHERE address NOT LIKE '%' || p.p\_location || '%';

1. For the schema shown in Tab 5.8, specify the following queries in relational algebra and in SQL
   1. For each flight, list the floight number, the departure airport for the first segment of the flight and the arrival airport for the last segment of the flight.

ρMINSEG 🡨Π min(segment\_no) FLIGHT\_SEGMENT

ρFLIG 🡨 FLIGHT  FLIGHT. flight\_no = FLIGHT\_SEGEMENT. flight\_no FLIGHT\_SEGMENT

ρSELBOK 🡨 AIRPORT AIRPORT.airport\_code = FLIG.DepartureSite\_code  FLIG

EXP1 🡨Π flight\_no, name (σ Segment\_no = minseg) (SELBOK)

ρMAXSEG 🡨Π max(segment\_no) FLIGHT\_SEGMENT

ρFLIG2 🡨 FLIGHT  FLIGHT. flight\_no = FLIGHT\_SEGEMENT. flight\_no FLIGHT\_SEGMENT

ρSELBOK2 🡨 AIRPORT AIRPORT.airport\_code = FLIG2.ArraivalSite\_code  FLIG2

EXP2 🡨Π flight\_no, name (σ Segment\_no = maxseg) (SELBOK2)

Result EXP1 UNION EXP2

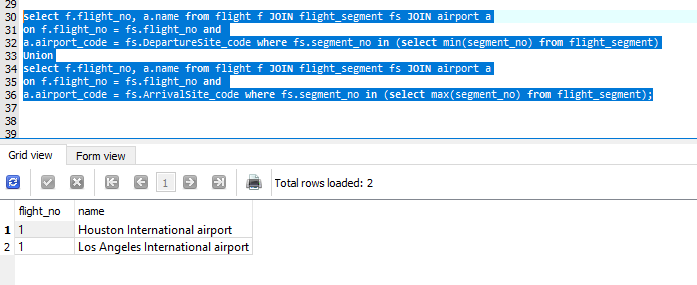
Highlighted expression is for sub query

select f.flight\_no, a.name from flight f JOIN flight\_segment fs JOIN airport a

on f.flight\_no = fs.flight\_no and a.airport\_code = fs.DepartureSite\_code where fs.segment\_no in (select min(segment\_no) from flight\_segment)

Union

select f.flight\_no, a.name from flight f JOIN flight\_segment fs JOIN airport a

on f.flight\_no = fs.flight\_no and a.airport\_code = fs.ArrivalSite\_code where fs.segment\_no in (select max(segment\_no) from flight\_segment); 

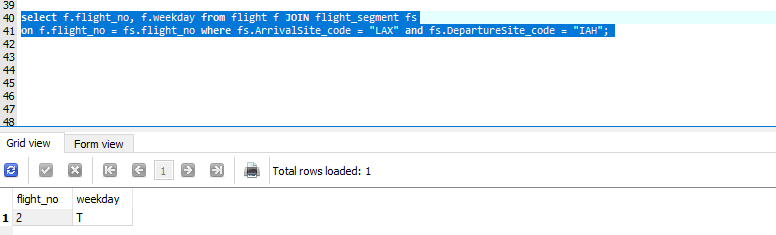
* 1. List the flight numbers and weekdays of all flights or flight segments that depart from Houston International airport (code = IAH) and arrive in Los Angeles International airport (code – LAX).

Note: I am assuming the highlighted part flights of flight segments

ρFLIG 🡨 FLIGHT  FLIGHT.flight\_no = FLIGHT\_SEGEMENT.flight\_no FLIGHT\_SEGMENT

Result 🡨Π flight\_no, weekday (σArrivalSite\_code = "LAX" and DepartureSite\_code = "IAH") (FLIG)

select f.flight\_no, f.weekday from flight f JOIN flight\_segment fs on f.flight\_no = fs.flight\_no where fs.ArrivalSite\_code = "LAX" and fs.DepartureSite\_code = "IAH";



* 1. List the flight number, departure airport code, scheduled departure time, arrival airport code, scheduled arrival time and weekdays of all flights or flight segments that depart from an airport located in the city of Houston, TX, and arrive at some airport  located in Los Angeles County, CA.

Note: I am assuming the highlighted part flights of flight segments

ρFLIG 🡨 FLIGHT  FLIGHT.flight\_no = FLIGHT\_SEGEMENT.flight\_no FLIGHT\_SEGMENT

ρSELAIR 🡨 AIRPORT AIRPORT.airport\_code = FLIG.DepartureSite\_code  FLIG

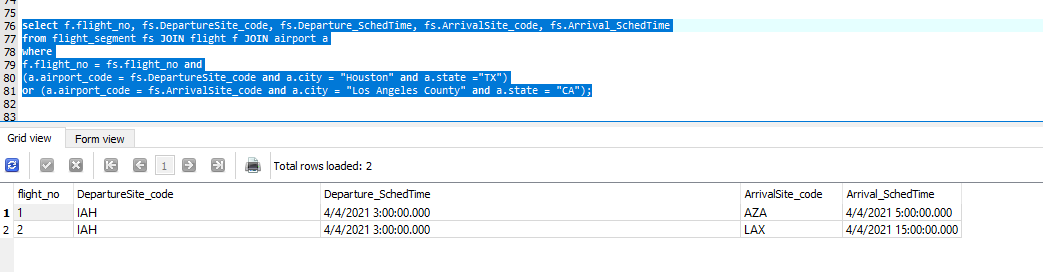
Result 🡨Π flight\_no, DepartureSite\_code, Departure\_SchedTime, ArrivalSite\_code, Arrival\_SchedTime (σ (airport\_code = DepartureSite\_code and city = "Houston" and state ="TX") or (airport\_code = ArrivalSite\_code and city = "Los Angeles County" and state = "CA")) (SELAIR)

select f.flight\_no, fs.DepartureSite\_code, fs.Departure\_SchedTime, fs.ArrivalSite\_code, fs.Arrival\_SchedTime

from flight\_segment fs JOIN flight f JOIN airport a

where f.flight\_no = fs.flight\_no and

(a.airport\_code = fs.DepartureSite\_code and a.city = "Houston" and a.state ="TX")or (a.airport\_code = fs.ArrivalSite\_code and a.city = "Los Angeles County" and a.state = "CA");



* 1. List all the fare information for flight CO197.

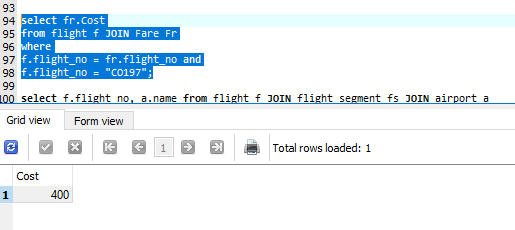
ρFLIG 🡨 FLIGHT  FLIGHT.flight\_no = FAIR.flight\_no FARE

Result 🡨Π cost (σflight\_no = "CO197") (FLIG)

select fr.Cost

from flight f JOIN Fare Fr

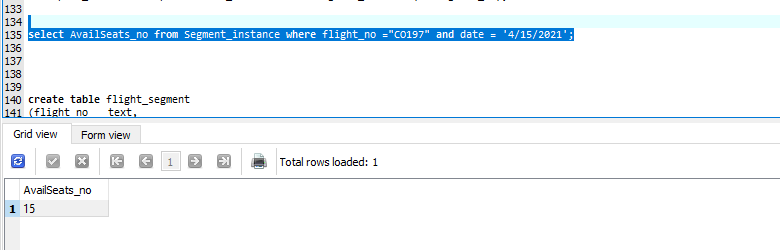
where f.flight\_no = fr.flight\_no and f.flight\_no = "CO197";



5.Retrieve the number of available seats for flight CO197 on the date 2021-4-15.

ΠAvailSeats\_no(σ(flight\_no ="CO197") and (date = “4/15/2021”)) (Segment\_instance)

select AvailSeats\_no from Segment\_instance where flight\_no ="CO197" and date = '4/15/2021';



1. For the relational schema shown in tab 6.12, assume that attribute names shared between tables represent PK-FK relationships. Write down the relational expressions and SQL for the following queries.
   1. How many copies of the book whose title is “The lost Tribe” are owned by the library branch in Sharpstown.

ρSELBOK🡨 (Library\_Branch  LIBRARY\_BRANCH.branch\_id = BOOKCOPIES.branch\_id BookCopies)

ρSELLIB 🡨 (Book  BOOK.isbn = SELBOK.isbn SELBOK)

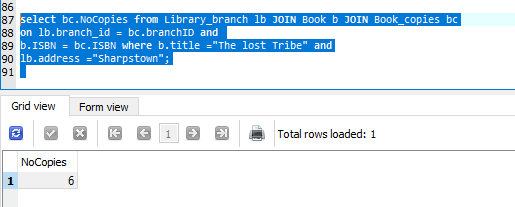
Result 🡨 ΠNoCopies(σ(title ="The lost Tribe") (and address ="Sharpstown")) (SELLIB)

select bc.NoCopies from Library\_branch lb JOIN Book b JOIN Book\_copies bc

on lb.branch\_id = bc.branchID and

b.ISBN = bc.ISBN where b.title ="The lost Tribe" and

lb.address ="Sharpstown";



* 1. How many copies of the book “Moby Dick” are owned by eaech library branch in the system?

ρSELBOK🡨 (Library\_Branch  LIBRARY\_BRANCH.branch\_id = BOOKCOPIES.branch\_id BookCopies)

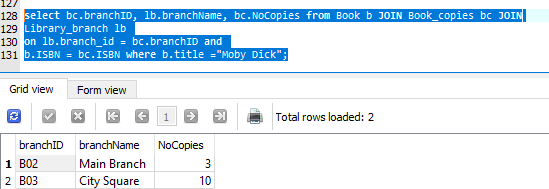
ρSELLIB 🡨 (Book  BOOK.isbn = SELBOK.isbn SELBOK)

Result 🡨 ΠbranchID, branchName, NoCopies(σ title ="Moby Dick") (SELLIB)

select bc.branchID, lb.branchName, bc.NoCopies from Book b JOIN Book\_copies bc JOIN

Library\_branch lb on lb.branch\_id = bc.branchID and b.ISBN = bc.ISBN

where b.title ="Moby Dick";



* 1. Retrieve the names of all borrowers who do not currently have any books checked out.

If dateDue is less thatn today’s date then the borrower doesn’t have any book or the borrower never took any book till now.

ρCRDNO 🡨 Πname(Book\_Loan)

ρSELBOK🡨 (BookBOOK.ISBN= BORROWER.ISBN Book\_Loan)

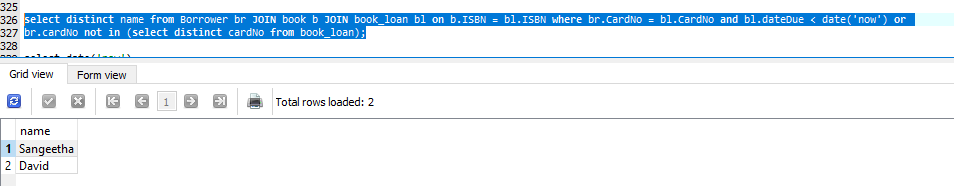
ρSELBOKLOAN🡨 Borrower  BORROWER.cardNo = SELBOK. cardNo SELBOK

Result 🡨 Πname (σdateDue < today\_date or cardNo != CRDNO) (SELBOKLOAN)

Highlighted expression is a sub query

select distinct name from Borrower br JOIN book b JOIN book\_loan bl on b.ISBN = bl.ISBN where br.CardNo = bl.CardNo and bl.dateDue < date('now') or

br.cardNo not in (select distinct cardNo from book\_loan);



* 1. For each book that is out on load from the Sharpstown branch and whose due data is 2021-3-24, retrieve the book title, the borrowers name and the borrowers address.

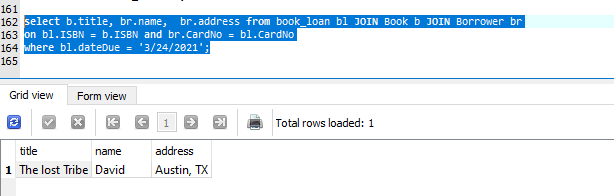
ρSELBOK🡨 (Book\_loan  BOOK.ISBN = BOOK\_LOAN.ISBN Book)

ρSELLIB 🡨 (Borrower  BORROWER.CardNo = SELBOK.CardNo SELBOK)

Result 🡨 Πtitle, name, address(σdateDue = '3/24/2021') (SELLIB)

select b.title, br.name, br.address from book\_loan bl JOIN Book b JOIN Borrower br

on bl.ISBN = b.ISBN and br.CardNo = bl.CardNo where bl.dateDue = '3/24/2021';



* 1. For each library branch, retrieve the branch name and the total number of unique book titles and the total number of books currently out on loan.

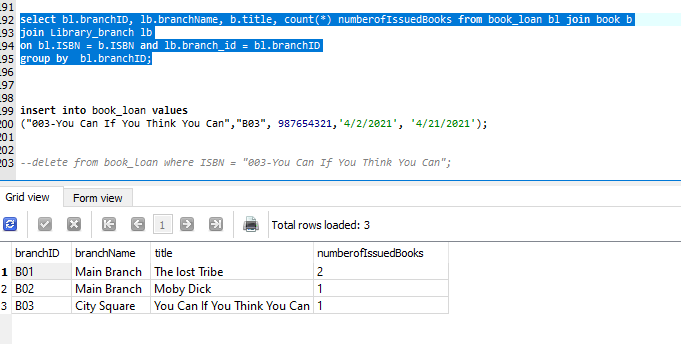
ρSELBOK🡨 (Book\_loan  BOOK.ISBN = BOOK\_LOAN.ISBN Book)

ρSELLIB 🡨 (Library\_Branch LIBRARY\_BRANCH.Branch\_Id = SELBOK. Branch\_Id SELBOK)

Result 🡨 Π branchID, branchName, title, COUNT(\*) (γbranchID) (SELLIB)

select bl.branchID, lb.branchName, b.title, count(\*) numberofIssuedBooks from book\_loan bl join book b join Library\_branch lb

on bl.ISBN = b.ISBN and lb.branch\_id = bl.branchID group by bl.branchID;



* 1. Retrieve the names, addresses and number of books checked out for all borrowers who current have five or more books checked out.

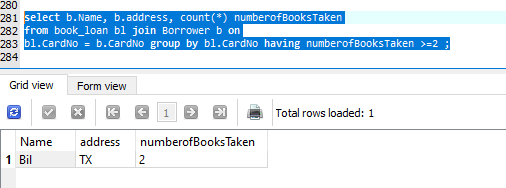
ρSELBOK🡨 (Book\_loan  BOOK.cardNo = BORROWER. cardNo Borrower)

Result 🡨 Π name, address, COUNT(\*) (γcount(\*) σ count(\*) > =2) (SELBOK)

select b.Name, b.address, bl.CardNo, count(\*) numberofBooksTaken

from book\_loan bl join Borrower b on

bl.CardNo = b.CardNo group by bl.CardNo having numberofBooksTaken >=2 ;



* 1. For each book authored by Jane Austen, retrieve the title and the number of copies of each owned by the library branch whose name is Central.

ρSELBOK 🡨 (Library\_Branch LIBRARY\_BRANCH.Branch\_Id = BOOKCOPIES.Branch\_Id BookCopies)

ρSELLIB 🡨 (Book  BOOK.isbn = SELBOK.isbn SELBOK)

ρSELLIB 🡨 (Book\_Authors  BOOK\_AUTHORS.isbn = SELLIB.isbn SELLIB)

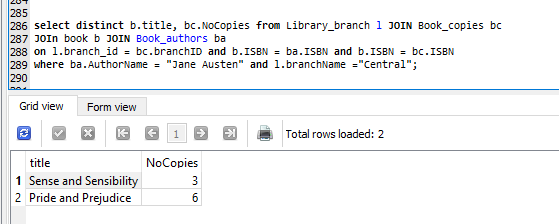
Result 🡨Π title, NoCopies (σ (AuthorName = "Jane Austen") and (branchName ="Central")) (SELIB)

select distinct b.title, bc.NoCopies from Library\_branch l JOIN Book\_copies bc

JOIn book b JOIN Book\_authors ba

on l.branch\_id = bc.branchID and b.ISBN = ba.ISBN and b.ISBN = bc.ISBN

where ba.AuthorName = "Jane Austen" and l.branchName ="Central";



5.For the schema specified in tab 5.13, specify the following queries in the relational algebra and in SQL.

2. List the order number and shipping data for all orders shipped from Warehouse A2.

ρSELORD 🡨 (Orders  ORDERS.cust\_no = CUSTOMER.cust\_no Customer)

ρSELSHIP 🡨 (Shipment  SHIPMENT.order\_no = SELORD.order\_no SELORD)

ρSELWARE 🡨 (Warehouse  WAREHOUSE.warehouse\_no = SELORD.warehouse\_no SELSHIP)

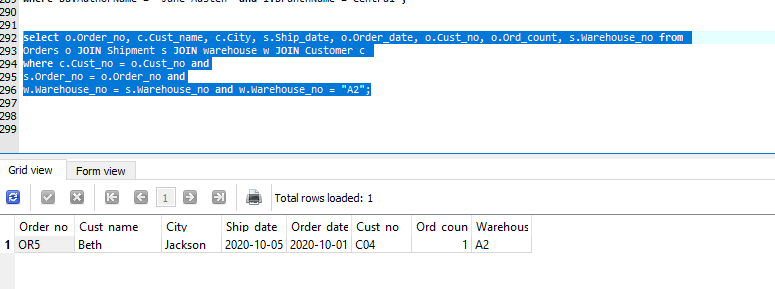
Result 🡨 Π Order\_no, Cust\_name, City, Ship\_date, Order\_date, Cust\_no, Ord\_count, Warehouse\_no (σWarehouse\_no = "A2") (SELWARE)

select o.Order\_no, c.Cust\_name, c.City, s.Ship\_date, o.Order\_date, o.Cust\_no, o.Ord\_count, s.Warehouse\_no from

Orders o JOIN Shipment s JOIN Warehouse w JOIN Customer c

where c.Cust\_no = o.Cust\_no and

s.Order\_no = o.Order\_no andw.Warehouse\_no = s.Warehouse\_no and w.Warehouse\_no = "A2";



1. List the warehouse information from which the Customer named James Menendez was supplied with his orders in the past 5 years – output should list the order number, date and warehouse number

ρSELORD 🡨 (Orders  ORDERS.cust\_no = CUSTOMER.cust\_no Customer)

ρSELSHIP 🡨 (Shipment  SHIPMENT.order\_no = SELORD.order\_no SELORD)

Result 🡨 Π Order\_no, Ship\_date, Warehouse\_no (σ Cust\_name = "James Menendez" and Ship\_date BETWEEN(CURRENT\_DATE, YEAR(CURRENT\_DATE) +5)) (SELSHIP)

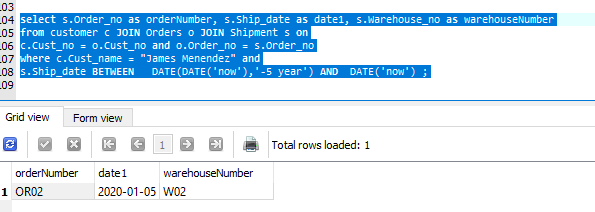
select s.Order\_no as orderNumber, s.Ship\_date as date1, s.Warehouse\_no as warehouseNumber

from customer c JOIN Orders o JOIN Shipment s on

c.Cust\_no = o.Cust\_no and o.Order\_no = s.Order\_no

where c.Cust\_name = "James Menendez" and

s.Ship\_date BETWEEN DATE(DATE('now'),'-5 year') AND DATE('now') ;



1. Produce a list that include the customer name, number of orders, the average count per order, with the columns in that order and sorted by ascending count.

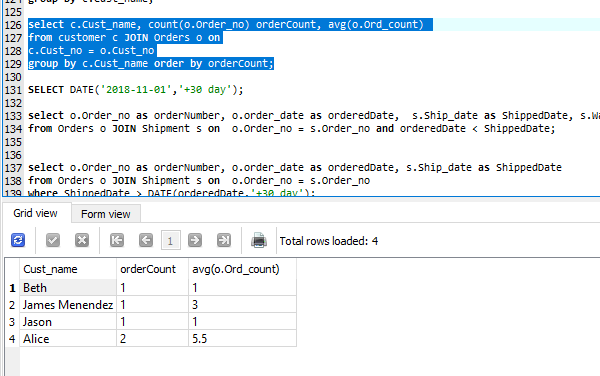
ρSELORD 🡨 (Orders  ORDERS.cust\_no = CUSTOMER.cust\_no Customer)

Result 🡨 **T** count(Order\_no) ( Π cust\_name, count(Order\_no), average(count(order\_no)) ( γcust\_name) ) ) (SELORD)

select c.Cust\_name, count(o.Order\_no) orderCount, avg(o.Ord\_count)

from customer c JOIN Orders o on

c.Cust\_no = o.Cust\_no group by c.Cust\_name order by orderCount;



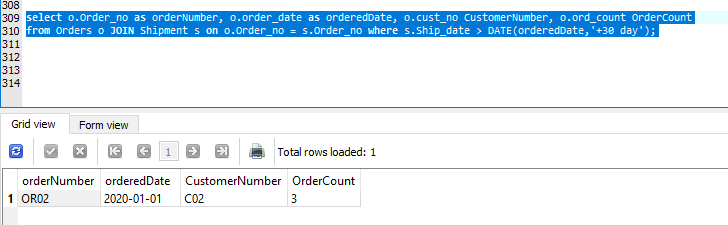
1. List the order that were not shipped within 30 days of the order being placed.

ρSELORD 🡨 (Orders  ORDERS.order\_no = SHIPMENT.order\_no Shipment)

Result 🡨 Π order\_no, order\_date, cust\_no, ord\_count (σ ship\_date > (order\_date+30 days)) (SELORD)

select o.Order\_no as orderNumber, o.order\_date as orderedDate, o.cust\_no CustomerNumber, o.ord\_count OrderCount

from Orders o JOIN Shipment s on o.Order\_no = s.Order\_no where s.Ship\_date > DATE(orderedDate,'+30 day');



1. List the order number for orders that were shipped from all the warehouses tha the company has in California.

ρSELORD 🡨 (Orders  ORDERS.order\_no = SHIPMENT.order\_no Shipment)

ρSELWAREH 🡨 (Warehouse  WAREHOUSE.warehouse\_no = SELORD.warehouse\_no SELORD)

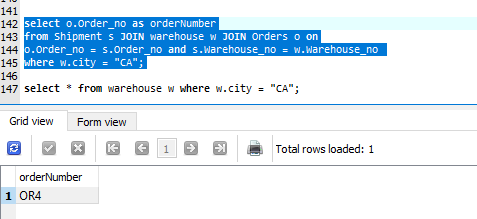
Result 🡨 Π order\_no(σ city=”CA”) (SELWAREH)

select o.Order\_no as orderNumber

from Shipment s JOIN warehouse w JOIN Orders o on

o.Order\_no = s.Order\_no and s.Warehouse\_no = w.Warehouse\_no

where w.city = "CA";



6. For the schema shown in tab 5.14, provide the queries as relational algebra and SQL.

1. Provide all of the attributes of the Trip table for trips whose total cost was greater than or equal to $2000.00

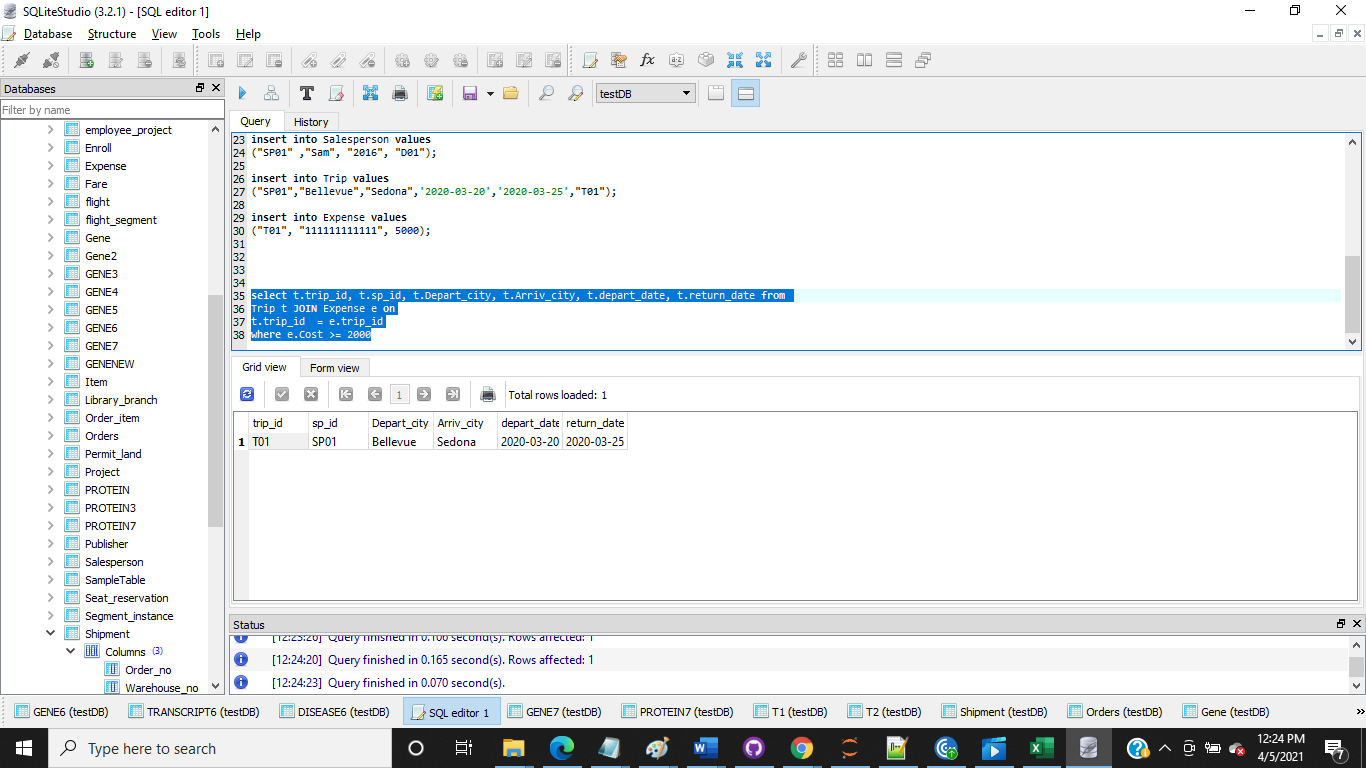
ρSELTRP 🡨 (Trip  TRIP.trip\_id = EXPENSE.trip\_id Expense)

Result🡨Π trip\_id, sp\_id, depart\_city,arriv\_city,depart\_date,return\_date(σ(cost>=2000)) (SELTRP)

select t.trip\_id, t.sp\_id, t.Depart\_city, t.Arriv\_city, t.depart\_date, t.return\_date from

Trip t JOIN Expense e on

t.trip\_id = e.trip\_id where e.Cost >= 2000



1. Provide the sales person ID for the individual(s) who took more than one trip to Honolulu in the space of a year.

ρSELSALESP 🡨 (Trip  TRIP.SP\_id = SALESPERSON. SP\_id Salesperson)

ρTRIPTHISYEAR 🡨 (σ(arriv\_city = "Honolulu") and (date\_diff(return\_date, depart\_date)<=365))(SELSALESP)

Result 🡨 Π SP\_id (γcount(trip\_id) σ count(trip\_id) > 1) (TRIPTHISYEAR)

select s.SP\_id from

Salesperson s JOIN Trip t on

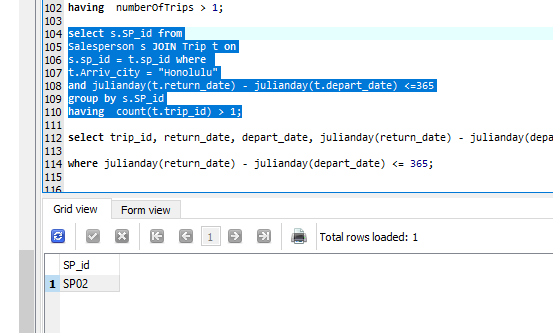
s.sp\_id = t.sp\_id where

t.Arriv\_city = "Honolulu"

and julianday(t.return\_date) - julianday(t.depart\_date) <=365

group by s.SP\_id

having count(t.trip\_id) > 1;



1. Print the total trip expenses incurred by the salesman whose id is S134

ρSELTRP 🡨 (Trip  TRIP.trip\_id = EXPENSE.trip\_id and TRIP.sp\_id = EXPENSE.sp\_id Expense)

ρSELCOS 🡨Π cost(σ(sp\_id=”S134”)) (SELTRP)

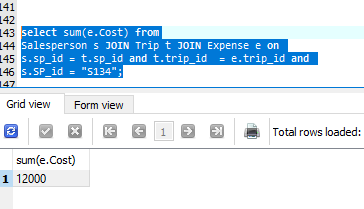
Result 🡨 **G**sum(cost)(SELCOS)

**G 🡪** Aggregate function

select sum(e.Cost) from

Salesperson s JOIN Trip t JOIN Expense e on

s.sp\_id = t.sp\_id and t.trip\_id = e.trip\_id and s.SP\_id = "S134";



1. For the schema given in tab 5.15, specify the following queries in the relational algebra and in SQL.
2. List the number of courses taken by all students named Brianna O’Conner in the Winter semester of 2020. Note that the usual coding would be S, U, W for the Spring, sUmmer and Winter semesters with a 2-digit designation for the year, such as S20.

ρSELSTUDENT 🡨 (Student  STUDENT.Student\_id = ENROLL.Student\_id Enroll)

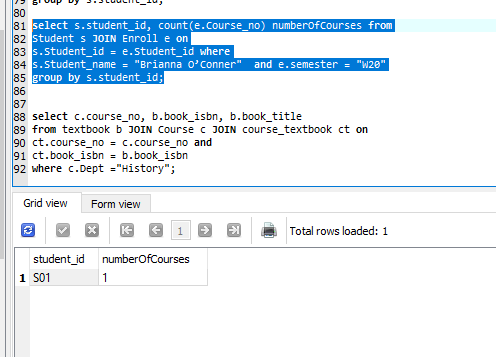
Result🡨Π Student\_id, count(Course\_no)(γcount(Student\_id) σ (Student\_name = "Brianna O’Conner" and (semester = "W20")) (SELSTUDENT)

select s.student\_id, count(e.Course\_no) numberOfCourses from

Student s JOIN Enroll e on

s.Student\_id = e.Student\_id where

s.Student\_name = "Brianna O’Conner" and e.semester = "W20" group by s.student\_id;



1. Produce a list of textbooks including the course number, book ISBN and book title for courses offered by the History department, where two or more books have been assigned to a course.

ρSELCOURSTB 🡨 (Course  COURSE.course\_no = COURSE\_TEXTBOOK.course\_no Course\_Textbook)

ρSELISBN 🡨 (TextBook  BOOK.book\_isbn = SELCOURSTB. book\_isbn SELCOURSTB)

ρHIST 🡨 (σ (dept = "History")(SELISBN)

ρSELCNO 🡨 Π course\_no (γcount(book\_isbn) σ count(book\_isbn) > 1) (Course\_Textbook)

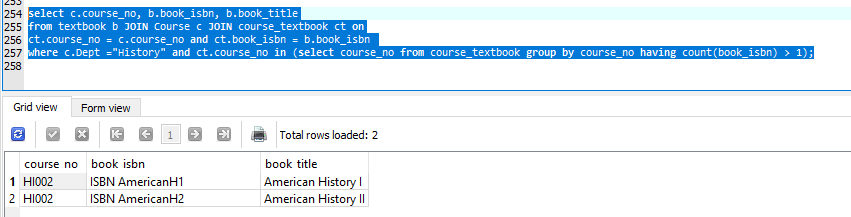
Result🡨Π course\_no, book\_isbn, book\_title(σ(SELCNO.course\_no = HIST.course\_no)) (SELCNO \* HIST)

Highlighted expression is for getting the value from sub query

select c.course\_no, b.book\_isbn, b.book\_title

from textbook b JOIN Course c JOIN course\_textbook ct on

ct.course\_no = c.course\_no and ct.book\_isbn = b.book\_isbn where c.Dept ="History" and ct.course\_no in (select course\_no from course\_textbook group by course\_no having count(book\_isbn) > 1);



2022. List any department and the name of the publisher,

where all of its adopted books are published by the same publisher.

ρSELCOURSTB 🡨 (Course  COURSE.course\_no = COURSE\_TEXTBOOK.course\_no Course\_Textbook)

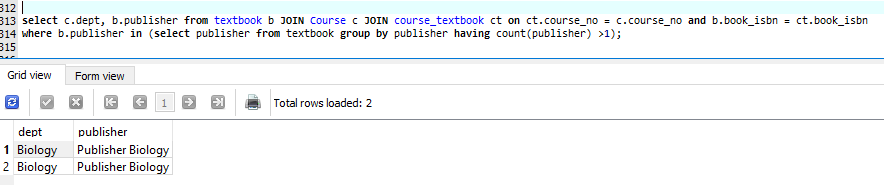
ρSELISBN 🡨 (TextBook  BOOK.book\_isbn = SELCOURSTB. book\_isbn SELCOURSTB)

ρSELPUBL 🡨 Π publisher (γcount(publisher) σ count(publisher) > 1) (Textbook)

Result🡨Π dept, publisher (σ(SELPUBL.publisher = SELISBN.publisher)) (SELPUBL \* SELISBN)

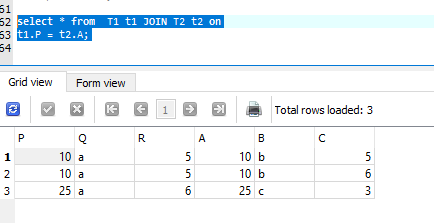
Highlighted expression is for getting the value from sub query

select c.dept, b.publisher from textbook b JOIN Course c JOIN course\_textbook ct on ct.course\_no = c.course\_no and b.book\_isbn = ct.book\_isbn where b.publisher in (select publisher from textbook group by publisher having count(publisher) >1);



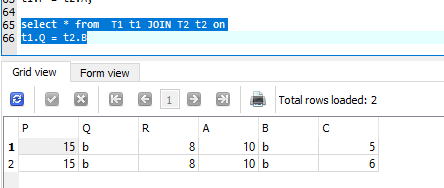
1. For the tables shown in tab 6.13, show the results of
   1. The inner join of T1 and T2 where T1.P = T2.A

select \* from T1 t1 JOIN T2 t2 on t1.P = t2.A;



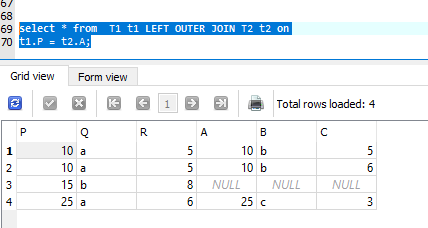
* 1. The inner join of T1 and T2 where T1.Q = T2.B

select \* from T1 t1 JOIN T2 t2 on t1.Q = t2.B



* 1. The left outer join of T1 and T2 where T1.P=T2.A

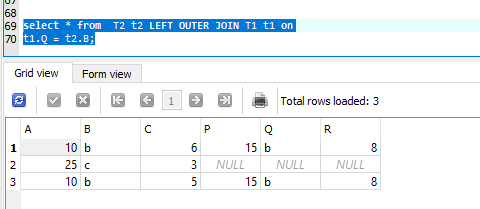
select \* from T1 t1 LEFT OUTER JOIN T2 t2 on t1.P = t2.A;



* 1. The right ourter join of T1 and T2 where T1.Q = T2.B

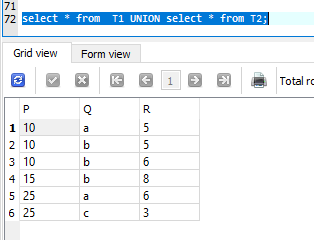
select \* from T1 t1 RIGHT OUTER JOIN T2 t2 on t1.P = t2.A; equivalent to

select \* from T2 t2 LEFT OUTER JOIN T1 t1 on t1.Q = t2.B;



* 1. The union of T1 and T2

select \* from T1 UNION select \* from T2;



* 1. The inner join of T1 and T2 where T1.P = T2.A AND T1.R = T2.C

select \* from T1 t1 JOIN T2 t2 where T1.P = T2.A AND T1.R = T2.C;

