**CIS 353 Notes**

**Week 2**

Query Languages:

Relational Algebra – procedural language

1. What we are looking for in table
2. How to get what we are looking for

Relational Calculus – non=procedural language

1. Only Tell you what we are looking for

Relational Algebra:

Fundamental –

Unary Operators: on single relation. Only outputs one table even with multiple outputs.

1. Selection - Unary
   1. () selections a subset of rows(horizontal)
   2. Grab values from the rows/tuples in the table
   3. Conditions are used along with columns
   4. Call: “ C > 3 (Table Name)
   5. Explained: Grab all tuples where column C in that tuple is greater than 3.
   6. OR/AND can be used to extend the conditions of the table search.
   7. )
   8. Explained: The first query will use the table from the results of the query in the parenthesis which uses the main Table to get its output.
   9. Original table-degree = Output table-degree
   10. Original table-cardinality >= Output Table-degree
2. Projection - Unary
   1. : Retains only desired columns
   2. Will be used to show output some attributes of a given(input) relation(table)
   3. Grab all values from column C, use the projection.
   4. Call:” : C, A (name of table)”
   5. Output will be a table of just columns C and A from the names table
   6. Will never have duplicate values in the table
   7. If more than one column the whole tuple would have to be the same as another tuple.
   8. Original table-degree >= Output table-degree
   9. Original table-cardinality >= Output Table-degree

Binary Operators: on pairs of relations (more than one table)

1. Cross-product – Binary
   1. {1, 2} x {3, 4}
   2. Now {{1, 3}, (1, 4), (2, 3), (2, 4)}
   3. Multiple each tuple in the first table by all tuples in the second.
   4. R1 degree + R2 degree = Odegree
   5. R1 card \* R2 card = Ocard
   6. (Book x Student)
   7. Places all the data in book and student in one table
   8. Find all names
   9. Add a condition then will sort out all information that the conditions effects.
   10. This has to be use when comparing two tables to find data
2. Union – Binary
   1. The tables must have same degree
   2. The attributes must have same domain-type
   3. R1 R2 will keep the degree of the original table but will add more tuples to the new table.
   4. PIE SID (Book) U PIE SID (OtherItems)
3. Set difference – Binary
   1. {1, 2, 3} and {1, 2, 5}
   2. Now {3}
   3. Only done on the tuples in tables
   4. Only displays what is left in the first table
   5. Original table-degree = Output table-degree
   6. Original table-cardinality >= Output Table-degree

Example:

Student(SID, Sname)

Book(Bid, Title, Sid)

OtherItems(Oid, Oname, Sid)

1. Find name of all Students
   1. PIE Sname,Sid (student)
      1. Need to have Sid do that they do not delete duplicate name

PIE Sid(Book) – PIE Sid(OtherItems)

* Project all of the Sid’s that have bough books but not other items

PIE sid(Students) - (PIE Sid(Book) – PIE Sid(OtherItems))

* Project all of the students who have only bought books

(PIE sid(Students) - PIE Sid(Book)) – PIE Sid(OtherItems))

* Project all students who have not bought anything

Find the names of all students that have a book

PIE student.Sname(6 book.sid = student.sid(book x student)

* book.sid = student.sid, this condition looks to make sure that when multiplying tables together it is only showing data where the Sid’s match up.

Student(SID, Sname)

Book(Bid, Title, Sid)

Buy(Bid, Sid, DOP)

Publisher(Pid, Pname, Bid, DOP)

Find the book title published by ‘ABC’

* PIEBook.title(6 book.bid = publisher.bid ^ publisher.Pname = ‘ABC’(Book x Publisher))

Find the book tittles that was bought by student ‘ABC’ and on January 2024

* PIE (6 Student.sid = book.sid ^ Student.Sname = ‘ABC’ ^ book.bid = buy.sid ^ student.sid = buy.sid (Student x Book)