Relational Algebra
Part Two

Database Management - CIS 386 01 FA17

By: Dr. Aos Mulahuwaish

Join Operations

- Join is a derivative of Cartesian product.
- Equivalent to performing a Selection, using join predicate as selection formula, over Cartesian product of the two operand relations.
- One of the most difficult operations to implement efficiently in an RDBMS and one reason why RDBMSs have intrinsic performance problems.

Join Operations

- Various forms of join operation
  - Natural join
  - Outer join

Natural join

- $R \bowtie S$
  - An Equijoin of the two relations R and S over all common attributes *x*. One occurrence of each common attribute is eliminated from the result.

Example - Natural join

- List the names and comments of all clients who have viewed a property for rent.

  $(\Pi_{\text{clientNo, fName, lName}}(\text{Client})) \bowtie$
  $(\Pi_{\text{clientNo, propertyNo, comment}}(\text{Viewing}))$

clientNo	fName	lName	propertyNo	comment
CR76	John	Kay	PG4	too remote
CR56	Aline	Stewart	PA14	too small
CR56	Aline	Stewart	PG4	
CR56	Aline	Stewart	PG36	
CR62	Mary	Tregear	PA14	no dining room

Outer join

- To display rows in the result that do not have matching values in the join column, use Outer join.

- R ⟕ S
  - (Left) outer join is join in which tuples from R that do not have matching values in common columns of S are also included in result relation.

Example - Left Outer join

- **Produce a status report on property viewings.**

$\Pi_{\text{propertyNo, street, city}}(\text{PropertyForRent}) ⟕ \text{Viewing}$

propertyNo	street	city	clientNo	viewDate	comment
PA14	16 Holhead	Aberdeen	CR56	24-May-01	too small
PA14	16 Holhead	Aberdeen	CR62	14-May-01	no dining room
PL94	6 Argyll St	London	null	null	null
PG4	6 Lawrence St	Glasgow	CR76	20-Apr-01	too remote
PG4	6 Lawrence St	Glasgow	CR56	26-May-01	
PG36	2 Manor Rd	Glasgow	CR56	28-Apr-01	
PG21	18 Dale Rd	Glasgow	null	null	null
PG16	5 Novar Dr	Glasgow	null	null	null

Division

- $R \div S$
  - Defines a relation over the attributes C that consists of set of tuples from R that match combination of *every* tuple in S.

- Expressed using basic operations:

  $T_1 \leftarrow \Pi_C(R)$

  $T_2 \leftarrow \Pi_C((S \times T_1) - R)$

  $T \leftarrow T_1 - T_2$

Example - Division

- Identify all clients who have viewed all properties with three rooms.

$$(\Pi_{\text{clientNo, propertyNo}}(\text{Viewing})) \div (\Pi_{\text{propertyNo}}(\sigma_{\text{rooms = 3}}(\text{PropertyForRent})))$$

$\Pi_{\text{clientNo,propertyNo}}(\text{Viewing})$

clientNo	propertyNo
CR56	PA14
CR76	PG4
CR56	PG4
CR62	PA14
CR56	PG36

$\Pi_{\text{propertyNo}}(\sigma_{\text{rooms=3}}(\text{PropertyForRent}))$

propertyNo
PG4
PG36

RESULT

clientNo
CR56

Aggregate Operations

- $\mathfrak{I}_{AL}(R)$
  - Applies aggregate function list, AL, to R to define a relation over the aggregate list.
  - AL contains one or more (<aggregate_function>, <attribute>) pairs .
- Main aggregate functions are: COUNT, SUM, AVG, MIN, and MAX.

Example – Aggregate Operations

- How many properties cost more than £350 per month to rent?

$\rho_R(\text{myCount})\ \Im_{\text{COUNT propertyNo}}\ (\sigma_{\text{rent} > 350}$ $(\text{PropertyForRent}))$

myCount
5

(a)