Learning Objectives

- After this segment, students will be able to
  - Describe RECURSIVE statement in SQL3
  - Use it to query Graphs

Querying Graphs: Overview

- Relational Algebra
  - Can not express transitive closure queries

- Two ways to extend SQL to support graphs
  1. Abstract Data Types
  2. Custom Statements
     - SQL2 - CONNECT clause(s) in SELECT statement
     - SQL3 - WITH RECURSIVE statement

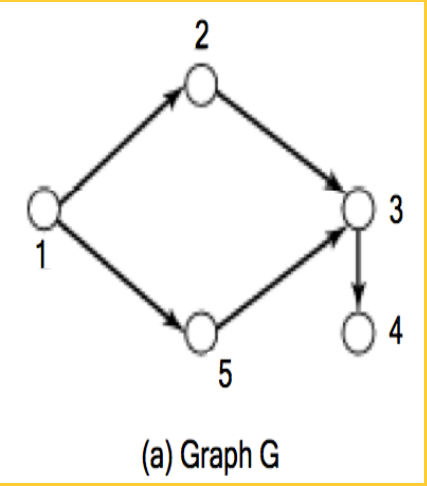
WITH RECURSIVE: Input, Output

- Input:
  - (a) Edges of a directed graph G
  - (b) Sub-queries to
    - Initialize results
    - Recursively grow results
    - Additional constraints

R

SOURCE	DEST
1	2
1	5
2	3
3	4
5	3

(b) Relation form



(a) Graph G

- Output: Transitive closure of G
  - Ex. Predecessors of a node
  - Ex. Successors of a node

Syntax of WITH RECURSIVE Statement

```
WITH RECURSIVE X(source,dest)
AS (SELECT source,dest FROM R )
        UNION
(SELECT R.source,  X.dest
        FROM R,  X
        WHERE R.dest=X.source )
```

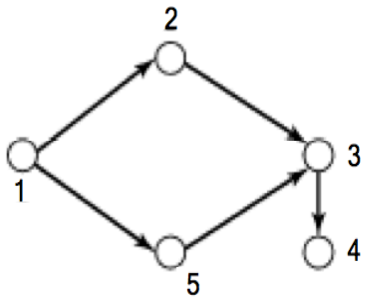
Description of Result Table

Initialization Query

Recursive Query to grow result

Example Input and Output

```
WITH RECURSIVE  X(source,dest)
AS  (SELECT source,dest FROM R )
            UNION
      (SELECT R.source,  X.dest
            FROM R,  X
            WHERE R.dest=X.source )
```



(a) Graph G

R

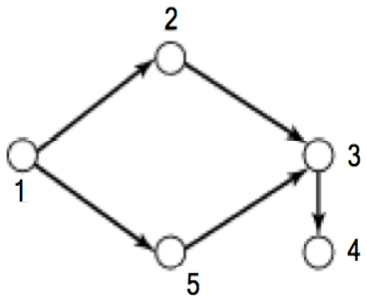
SOURCE	DEST
1	2
1	5
2	3
3	4
5	3

(b) Relation form

(c) Transitive closure (G) = Graph G

(d) Transitive closure in relation form

SQL3 Recursion Example - Meaning



(a) Graph G

R

SOURCE	DEST
1	2
1	5
2	3
3	4
5	3

(b) Relation form

X

SOURCE	DEST

(c) Transitive closure (G) = Graph G

(d) Transitive closure in relation form

- Initialize X by
  (SELECT source,dest FROM R)
- Recursively grow X by
  (SELECT R.source, X.dest
  FROM R, X
  WHERE R.dest=X.source)
- Infer X(a,c) from R(a,b),X(b,c)
- Infer X(1,3) from R(1,2),X(2,3)
- Infer X(2,4) from R(2,3),X(3,4)
- Infer X(5,4) from R(5,3),X(3,4)
- Infer X(1,4) from R(1,5),X(5,4)