DSE 203

DAY 1: REVIEW OF DBMS CONCEPTS

Data Models

- A specification that precisely defines
  - The structure of the data
  - The fundamental operations on the data
  - The logical language to specify queries on the data
- Example
  - Relational
  - Array-structured
  - Tree-structured
  - Graph-structured
  - Vector-structured
- Three architectural levels
  - Conceptual Model
  - Logical Model
  - Physical Model

Database Schema

- Organization of Data for a Specific Application
- Based on a Data Model
- Specific Integrity Constraints
  - Key constraint, uniqueness constraint, …
- Can represent an infinite number of database instances
- Many databases do not have to have a schema
  - XML, JSON databases
  - Graph Databases
  - Having a schema is useful for query formulation and for query evaluation

Relational Data Model

- Fields or attributes
  - Atomic or complex
- Domain of an attribute
- Tuple or record of attributes
- Relation = Set of tuples
- The special value called NULL
- Set and bag (multiset) semantics

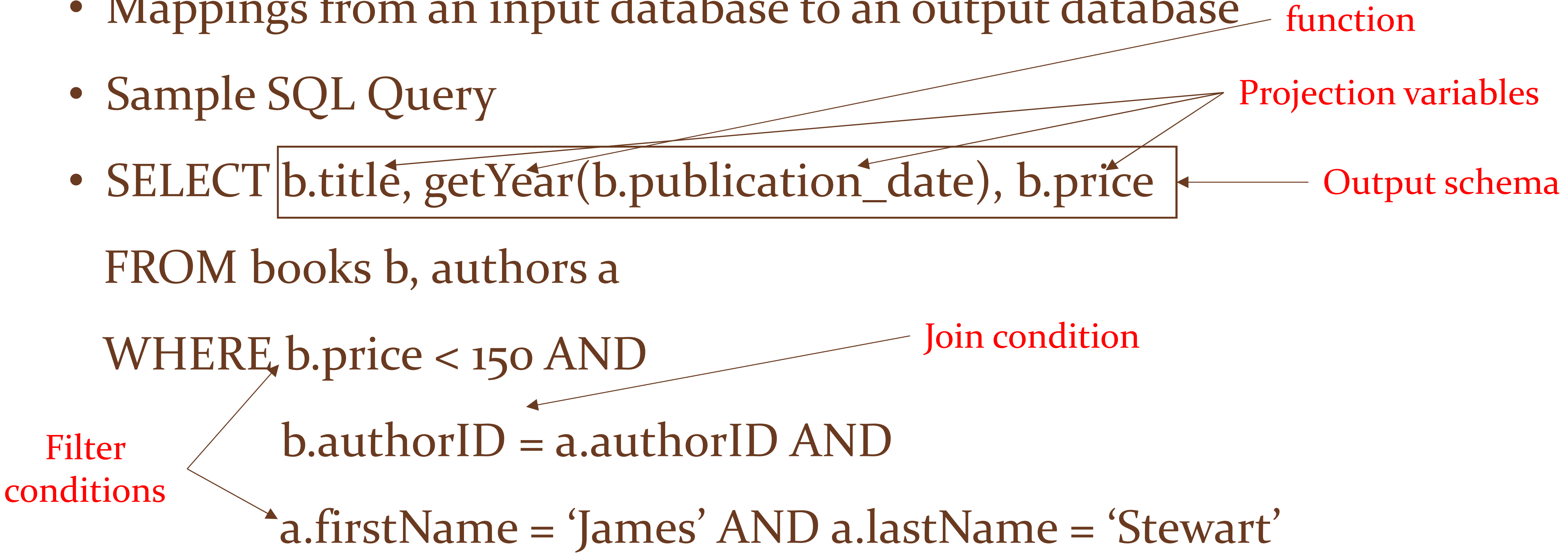
- Relational Algebra Operators
  - Selection
  - (generalized) Projection
  - Cross product
  - Joins (inner join, outer join, semioin, …)
  - Union
  - Difference
  - Rename
- Other operations
  - Group By
  - Aggregates

Integrity Constraints in Relational Databases

- Key Constraint
  - Let **A** be the set of attributes of a relation *R*
  - $S \subseteq$ **A** such that if t_1, t_2 are tuples in R then, if $t_1^S = t_2^S$, then $t_1 = t_2$
  - Then S is key of R
- Functional Dependency
  - Let **A** be the set of attributes of a relation *R*
  - S, S’ $\subseteq$ **A** such that if t_1, t_2 are tuples in R and t_1^S etc. represent subtuples with attribute set S, then, if $t_1^S = t_2^S$ then $t_1^{S'} = t_2^{S'}$ for every pair of tuples
  - Then S’ is functionally determined by S

Queries

- Mappings from an input database to an output database
- Sample SQL Query



- SELECT b.title, getYear(b.publication_date), b.price
  FROM books b, authors a
  WHERE b.price < 150 AND
  b.authorID = a.authorID AND
  a.firstName = ‘James’ AND a.lastName = ‘Stewart’

function

Projection variables

Output schema

Join condition

Filter conditions

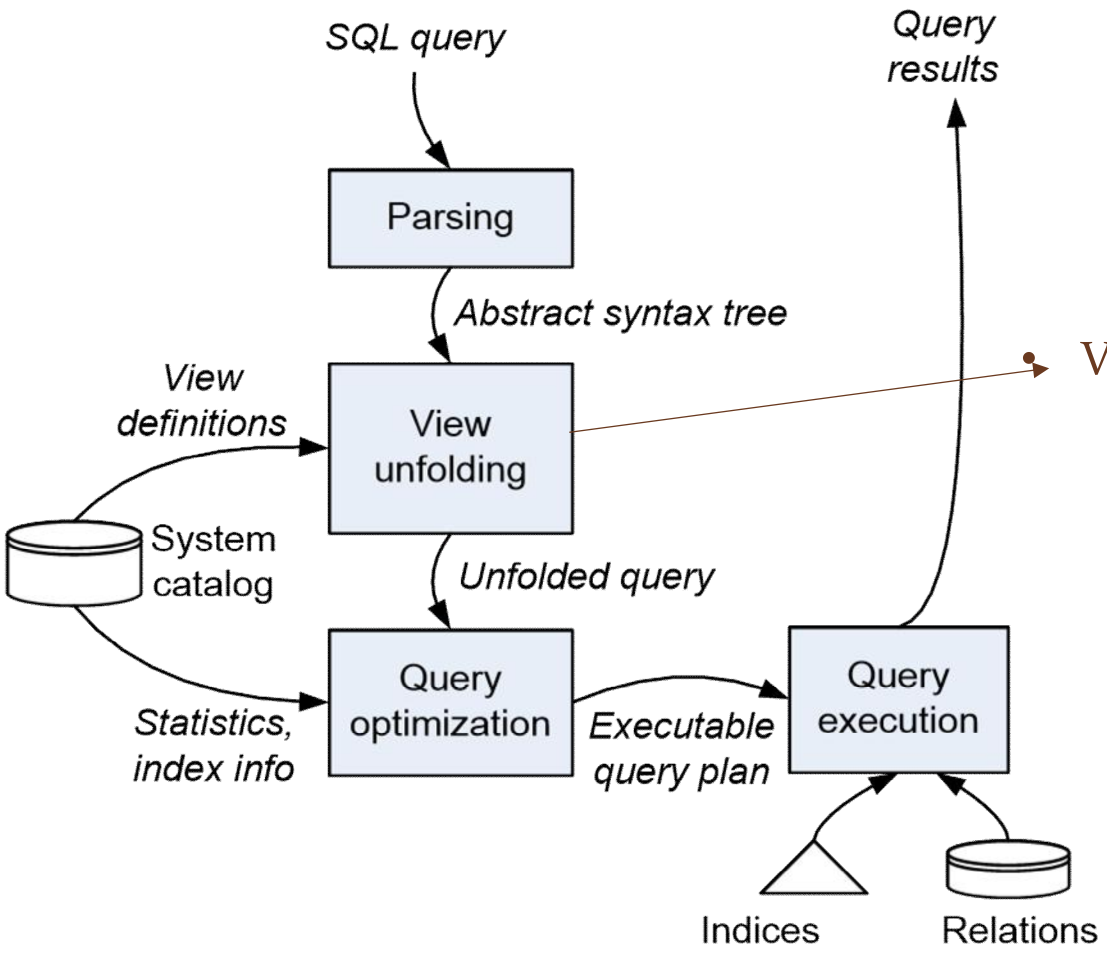
Logic-like Representation

- Schema
  - Books(Title, Author_id, ISBN, Publication_year, Price)
  - Authors(Author_id, FirstName, LastName)
- Fact tuples
  - Book(‘Operating System Concepts’, 438, ‘978-1118063330’, 2012, 134.36)
  - Authors(438, ‘Abraham’, ‘Silberschatz’)
- Query
  - Result(X,Y,Z):- Books(X, **A**, ISBN, P, Z), Authors(**A**, F, L), Z < 150, F= ‘James’, L=‘Stewart’, Y = getYear(P)
  - Result(X,Y,Z):- Books(X, **A1**, ISBN, P, Z), Authors(**A2**, F, L), Z < 150, F= ‘James’, L=‘Stewart’, Y = getYear(P), **A1 = A2**

Relational Views

- Named Virtual Relation
- Defined as a query to a set of *base tables or on other views*
- Use
  - Only need a specific subset of the data for an application
- Example
  - CREATE VIEW V AS <*query expression*>
- Materialized View
  - An actual table corresponding to the view definition is created
  - This table is maintained as the base tables get updated

Query Evaluation in a DBMS



- View Unfolding
  - (Recursively) replacing a view with its definition until the query is fully expressed against the base tables

Distributed Query Processing

Suppose our data is distributed across multiple machines and we need to process queries

- **Parallel DBMSs** assume homogeneous nodes and fast networks (sometimes even shared memory)
    - A Major goal: efficiently utilize all resources, balance load
- **Distributed DBMSs** assume heterogeneous nodes, slower networks, some sources only available on some machines
    - A Major goal: determine what computation to place where

➢Our focus here is on the latter

Distributed Query Processing

- Data Placement
  - Horizontal partitioning
  - Vertical partitioning
  - Hybrid Partitioning
- Data Shipping
  - Ship operation – sending the output of a query from one machine to another
  - Exchange operation – exchanges tuples across a set of data nodes of a horizontally partitioned database until all data with the same key are co-located
    - A suitable partitioning function is used

Distributed Query Processing

- 2 phase joins
  - Two relations are on two machines and the query needs to join them
  - Compute a summary (e.g., projection) of the join attributes from one relation and ship to the second
  - The second machine returns performs a local join and forms a partial result structure to the first machine
  - The first machine completes the join by using these tuples
- What is this kind of operation called?
- What happens when even the summary is really large?