Introduction to Data Management
CSE 344

Lecture 27: Final Review

Final Exam

- Tuesday, March 18, 2:30-4:30 pm, in class
- Content: **Everything**
  - But focus on Lectures 16 through 27 more
- Open books and open notes
  - You can bring **any notes** that you want including slides (check for updates), assignments, old exams, stuff from the web, …
  - BUT NO laptops, phones, or other devices
- Additional office hour (sudeepa)

  cse 303, 3/15(Sat), 2 – 4 pm

How To Study

- Go over the lecture notes
- Read the book
- Go over the homeworks (your graded solution + ours)
- Practice (within stipulated time)
  - Practice web quiz (entire question explanation visible now)
  - Finals from past 344
  - Look at both midterms and finals from 444 past years: be careful because several questions do not apply to us!
  - Questions in the book
  - Some old finals have 100 points, some 200 points, budget your time
- **Ask course staff questions on discussion board**
- The goal of the final is to help you learn

Final Exam

- 110 mins, X points in total => On average (110/X) min/point
- Some questions will take more time, some less.
- Make sure you do not spend too much time on any single problem (considering all its subparts).
- Take a quick look at the entire exam and see what can be done easily and fast.
- Questions can be conceptual or problem solving , write short answers (few words are fine).
- Write partial solutions for partial credit even if you did not get to the final answer

Today: Final Review

What did we learn?

1. Relational model, SQL (lectures 1-8)
2. RA/RC/Datalog (lectures 9-12)
3. XML (lectures 13-14)
4. Database design – ER Diagram, Constraints, Normalization (lectures 15-18)
5. Views (lecture 19)
6. Transactions (lecture 20-21)
7. Parallel Databases, MapReduce (lecture 22-25)
8. Data Integration, NoSQL (lecture 26)

1. SQL

Know the syntax

- SELECT-FROM-WHERE
- DISTINCT, ORDER BY, renaming of attributes
- CREATE TABLE, KEY, REFERENCES, INSERT, DELETE, UPDATE
- GROUP BY, HAVING
- NULLs (3-valued logic), outer joins
- Nested queries (subqueries): (NOT) EXISTS, (NOT) IN, EXCEPT, ANY, ALL
  - notice when we need to have the same schema

Know the semantics

- nested loop
- in which order the clauses are evaluated

1. SQL

Remember

- SELECT attributes are aggregates or must appear in GROUP BY
- WHERE (condn on tuples) vs. HAVING (condn on groups)
- HAVING condition should be well-defined for groups (may or may not involve aggregates)
  - –non-aggregate attributes must appear in GROUP BY

Indexes (use, types, standard implementation)

2. RA/Datalog/RC

- Relational algebra
  - Operators: $\cup$, $\cap$, -, σ, Π, $\times$, $\bowtie$, ρ, δ, γ, τ
  - Some of them need same schema R – S, R $\cup$ S, etc
  - Joins: natural-, equi-, theta-
  - Easier to write bottom-up as a logical query plan
  - Logical vs. Physical query plan
  - Logical and Physical data independence

2. RA/Datalog/RC

- Datalog (+ negation, - recursion)
  - Head, body, atom/subgoal, head/existential variables
  - How to express union, intersection, selection, projection, join, difference
  - Safe vs. unsafe rules: datalog rules must be safe!

- Relational Calculus
  - all variables that do not appear in the final answer must be quantified
  - "some", "exist(s)" (sufficient condition): ∃
  - "all", "none exist" (necessary condition): ∀… ⇒

2. RA/Datalog/RC

- Important translations:
  - RC → SQL
  - RC → Datalog
  - RC → RA
  - SQL → RA
  - Datalog → SQL

  - Often convenient to first translate to datalog

3. XML

- Basic syntax: elements, attributes; well-formed vs. valid document
- Xpath (/.. vs. //..)
- XQuery (predicates, position, count, ….)

3. XML

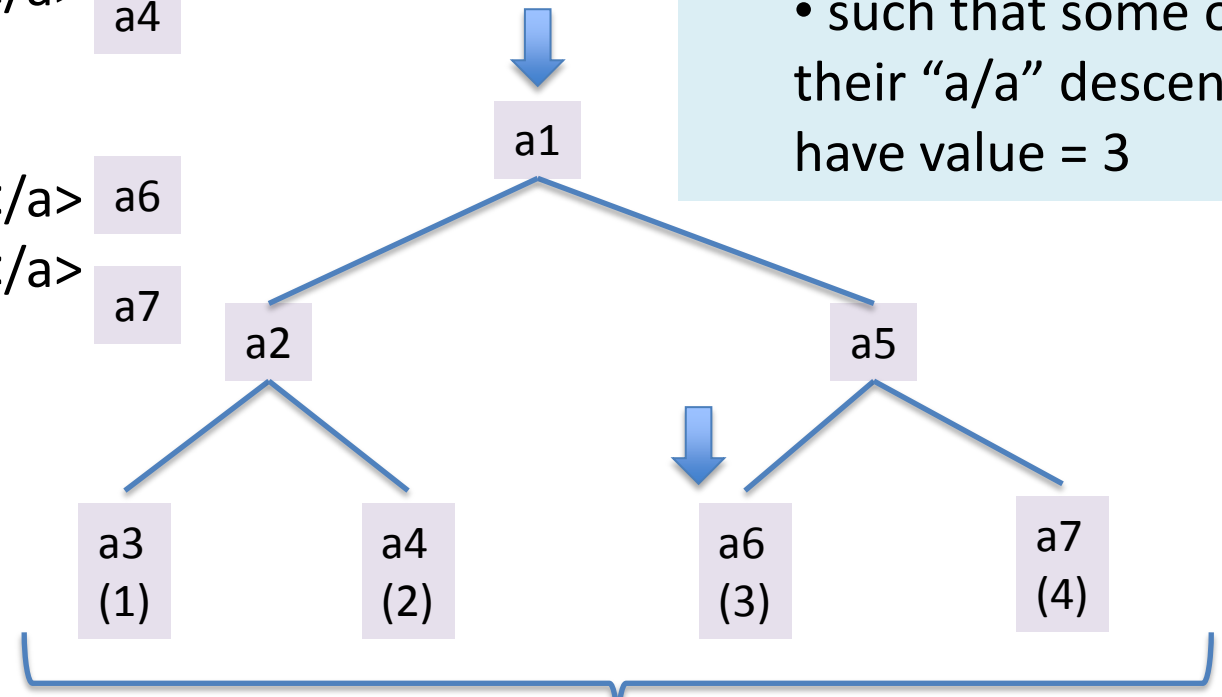
- Sometimes convenient to think in terms of a "tree"

```
<a>  a1
    <a>  a2
        <a> 1 </a>  a3
        <a> 2 </a>  a4
    </a>
    <a>  a5
        <a> 3 </a>  a6
        <a> 4 </a>  a7
    </a>
</a>
```

Q. How many elements in
//a[a/a/text()=3]/a/a

=
- find "a/a" descendants
- of all "a" element

at any depth
- such that some of their "a/a" descendants have value = 3



Ans = 4

4. Database Design

E/R diagrams:

- Entities, attributes
- Relationships:
  - Many-many, many-one, one-one, exactly one
  - Multi-way relationships (what do multiple arrows mean?)
- Inheritance, weak entity sets, union types
- Constraints in E/R diagrams
- Practice:
  - Identifying keys, attribute sets
  - Translation to relations

4. Database Design

Constraints in SQL

- Keys and Foreign Keys
- Referential integrity
  - on update/delete reject, cascade, set null
- Attribute/Tuple-based constraints
  - Predicates on values
  - NOT NULL
- Constraints vs. Assertion vs. Trigger

4. Database Design

Conceptual Design

- Data anomalies
- Functional dependencies
  - Make sure you can check if a table satisfies a set of FDs
- Keys and Super keys
- BCNF

Algorithms to practice:

- Attribute closure
- Decomposition to BCNF (identify keys after decomposition)
- Chase Test (to test if a given decomposition is lossless)

5. Views

- Types of views: virtual v.s. materialized views
- Definition and how to use them
- CREATE VIEW in SQL
- Query modification
  - query using views -> efficient equivalent SQL queries
- How to rewrite queries that will only use the views
- Horizontal vs. Vertical partitioning

6. Transactions

Transactions concepts

- ACID properties
- *Serializability (for Isolation), Conflicts (RW, WR, WW, actions of same Transaction)*
- Schedules:
  - serial, serializable, conflict-serializable, recoverable (need for strict 2PL)
- Concurrency control using locks
  - SQLite and SQLServer examples
- Phantoms
- Deadlocks
- Transactions in SQL :
  - Four isolation levels
  - commit/rollback

6. Transactions

Algorithms/approaches to review

- Precedence graphs
  - to check conflict serializability, conflict equivalence(WQ!)
- Lock transitions in SQLite
- Where to insert L(A), U(A)
  - for 2PL, strict 2PL
  - sometimes you need WAITING…. GRANTED

6. Transactions

READ UNCOMMITTED

T1	T2	A
		10
r1(A) = 10		
	L2(A) w2(A, 20)	
r1(A) = 20		
	U2(A) A2	

- Dirty Read
- No Read lock
- Strict 2PL write lock

Note: simplified model.
SQL Server uses shared/update/ exclusive and other locks

READ COMMITTED

T1	T2	A
		10
SL1(A) r1(A) = 10 SU1(A)		
	L2(A) w2(A, 20)	
	U2(A) C2	20
SL1(A) r1(A) = 20 SU1(A)		

- Unrepeatable (but not dirty) Read
- Short-term Read lock
- Strict 2PL write lock

REPEATABLE READ

T1	T2	A
		10
L1(A) r1(A) = 10		
	L2(A) WAITING	
r1(A) = 10		
U1(A) C1		
	L2(A) GRANTED w2(A, 20)	
	U2(A) C2	20

- Repeatable Read
- Strict 2PL Read lock
- Strict 2PL write lock

7. Parallel Data Processing

Parallel databases:

- Speedup/scaleup
- Shared memory, shared disk, shared nothing
- Horizontal data partition: block, hash, range
  - Notice the relevant attribute for hash/range
  - Skew
  - Which tuples stay together
- How to implement simple algorithms: group-by, join
- How to execute a complete query in parallel (parallel query plan)
  - Leverage parallelism as much as possible
  - Write and describe (e.g. hash) shuffle phases

7. Parallel Data Processing

MapReduce

- Functions: map, (combine), reduce
- Terminology: MapReduce job; Map/Reduce task; Map/Reduce function; Nodes
- Basic implementation of MR
- Dealing with server failures and stragglers
- How to express simple computations in MapReduce
  - Some queries may need multiple MR phases
  - You will not be asked to write a Pig Latin query

8. Data Integration, NoSQL

- Why do we need these?
- Standard approaches
- Challenges

More to learn in 444!

Algorithms/implementation

1. Data storage and buffer management
2. Implementation and use of indexes
3. (cost-based) Query optimization
4. Transaction : concurrency control and recovery
5. Distributed query processing
6. and more...

Thank you and good luck!