Introduction to Database Systems
CSE 414

Lecture 30: Final Review

We're almost done!

- HW8 due tonight – **NO LATE DAYS**
  – Please be sure to shut down all jobs and check your account charges
  – Sample solution posted Saturday
- Final exam: Monday, 2:30, here
  – Review Q&A Sunday, 2 pm, SAV 260
  – Covers everything but biased towards things since midterm (e.g., XML onward)
  – Closed book, no notes
    - Reference material included in exam as needed
- Course summary today

How to Study for the Final

- Go over the lecture notes
- Read the book
- Go over the assignments
- Practice
  - Finals from past 414s and 344s
  - Questions in the book
- The goal of the final is to help you learn!

The Final

Entire class content is on the final!

But focus of questions on the final will be as follows:

1. SQL and Relational Algebra + query plans (lectures 2-12)
2. XML (lectures 13-14)
3. Database design (lectures 15-19)
4. Views (lecture 20)
5. Transactions (lecture 21-23)
6. Parallel Databases (lecture 24-29)

1. SQL including Views

SQL

- SELECT-FROM-WHERE
- DISTINCT, ORDER BY, renaming of attributes
- INSERT, DELETE, UPDATE
- GROUP-BY and HAVING: *different* from WHERE (why ?); restriction on attributes and aggregates in select
- NULLs, outer joins
- Nested queries (subqueries)

Know the syntax

Know the semantics (nested loops)

1. SQL and Relational Query Languages

SQL

- CREATE TABLE, plus constraints
- INSERT/DELETE/UPDATE

Indexing

- Clustered vs. unclustered
- Index selection problem

1. SQL and Relational Algebra

SQL = What, RA = How

- Union ∪, intersection ∩, difference –
- Selection σ
- Projection Π
- Cartesian product ×, join ⋈
- Rename ρ
- Duplicate elimination δ
- Grouping and aggregation γ

1. SQL and Relational Algebra

- Be able to translate SQL <-> RA
- Know basic ways of implementing query plans, particularly joins (nested loop, hash-join)
- Know how to compute costs of query plans (B(R), T(R), V(R,a), M)

2. XML

- Basic syntax: elements, attributes; well-formed vs. valid document
- XPath – basic navigation
- XQuery – complex queries
  - "The SQL of XML"
  - XPath expressions are simple XQueries

XML Terminology

Tags, Elements

Elements	**Attributes**
Ordered	Unordered
May be repeated	Must be unique
May be nested	Must be atomic

Document Type Definitions (DTD)

```
<!ELEMENT tag (CONTENT)>
```

content model

- Content model:
  - Complex = a regular expression over other elements
  - Text-only = #PCDATA
  - Empty = EMPTY
  - Any = ANY
  - Mixed content = (#PCDATA | A | B | C)*

XPath

bib matches a **bib** element

***** matches any element

/ matches the **root** element

/bib matches a **bib** element under **root**

bib/paper matches a **paper** in **bib**

bib//paper matches a **paper** in **bib**, at any depth

//paper matches a paper at any depth

paper|book matches a **paper** or a **book**

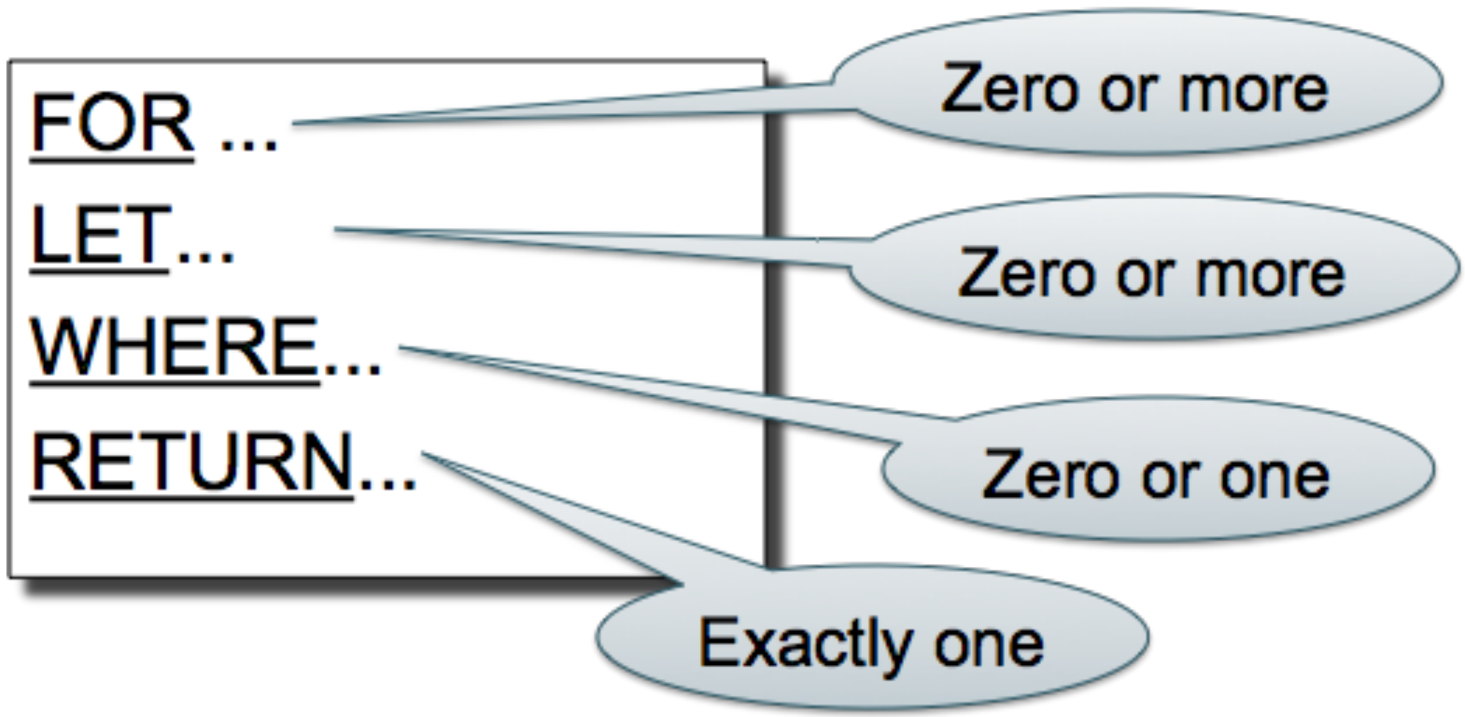
@price matches a **price** attribute

bib/book/@price matches price attribute in book, in bib

bib/book[@price<“55”]/author/last-name matches…

bib/book[@price<“55” or @price>”99”]/author/last-name matches…

Xquery



Nesting, distinct-values

3. Database Design

E/R diagrams:

- Entities, attributes
- Relationships:
  - Many-many, many-one, one-one, exactly one
  - Multi-way relationships
- Inheritance, weak entity sets, union types
- Constraints in E/R diagrams
- Translation to relations

3. Database Design

Constraints in SQL

- Keys and Foreign Keys
- Attribute level constraints
  - Predicates on values
  - NOT NULL

3. Database Design

Conceptual Design

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

Functional Dependency

A1 -> A2: If two tuples agree on the attribute A1 then they must also agree on the attribute A2

Closure:

Given a set of attributes $A_1, \ldots, A_n$

The **closure**, $\{A_1, \ldots, A_n\}^+$ = the set of attributes B s.t. $A_1, \ldots, A_n \rightarrow B$

Superkey

For all sets X, compute X+

- If X+ = [all attributes], then X is a superkey
- Try only the minimal X's to get the keys

Boyce-Codd Normal Form

There are no "bad" FDs:

> **Definition**. A relation R is in BCNF if:
>
> Whenever $X \rightarrow B$ is a non-trivial dependency, then X is a superkey.

Equivalently:

> **Definition**. A relation R is in BCNF if:
>
> $\forall$ X, either $X^+ = X$ or $X^+ =$ [all attributes]

BCNF Decomposition Algorithm

```
Normalize(R)
  find X s.t.: X ≠ X+ ≠ [all attributes]
  if (not found) then "R is in BCNF"
  let Y = X+ - X;      Z = [all attributes] - X+
  decompose R into R1(X ∪ Y) and R2(X ∪ Z)
  Normalize(R1);  Normalize(R2);
```

4. Views

- Types of views: virtual v.s. materialized views
- Definition and how to use them
- CREATE VIEW in SQL
- Query modification

5. Transactions

Transactions concepts

- Review ACID properties
- Definition of *serializability*
- Schedules, conflict-serializable and recoverable
- The four isolation levels in SQL
- Concurrency control using locks
  - SQLite and SQLServer examples
- Phantoms
- Deadlocks
- Transactions in SQL

ACID Properties

A DBMS guarantees the following four properties of transactions:

- Atomic
  - State shows either all the effects of txn, or none of them
- Consistent
  - Txn moves from a state where integrity holds, to another where integrity holds
- Isolated
  - Effect of txns is the same as txns running one after another (ie looks like batch mode)
- Durable
  - Once a txn has committed, its effects remain in the database

Serial / Serializable

- A *serial* schedule is one in which transactions are executed one after the other, in (some) sequetial order
- A schedule is *serializable* if it is equivalent to a serial schedule

Conflicts

Conflicts: pair of actions (in order) in schedule such that if swapped, then behavior changes.

Two actions by same transaction T_i: — $r_i(X); w_i(Y)$

Two writes by T_i, T_j to same element — $w_i(X); w_j(X)$

Read/write by T_i, T_j to same element — $w_i(X); r_j(X)$ — $r_i(X); w_j(X)$

Note: any # of actions can appear between them

Conflict Serializability

- A schedule is ***conflict serializable*** if it can be transformed into a serial schedule by a series of swaps of adjacent non-conflicting actions

- Stronger condition than serializability

- How do we check for conflict serializability?
  - Using Precedence Graph

Locking

- Two Phase Locking (2PL): In every transaction, all lock requests must precede all unlock requests

- Strict Two-Phase Locking (strict 2PL): All locks are held until the transaction commits or aborts.

Isolation Levels in SQL

1. "Dirty reads"
   SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

2. "Committed reads"
   SET TRANSACTION ISOLATION LEVEL READ COMMITTED

3. "Repeatable reads"
   SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

4. Serializable transactions
   SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

6. Parallel Data Processing

- Parallel databases:
- Speedup/scaleup
- Shared memory, shared disk, shared nothing
- Horizontal data partition: block, hash, range
- How to implement simple algorithms: group-by, join
- How to execute a complete query in parallel

6. Parallel Data Processing

MapReduce

- Functions: map, (combine,) reduce
- Terminology: chunk, map job / reduce job; map task / reduce task; server (instance); failed server
- Basic implementation of MR
- Dealing with server failures and stragglers
- How to express simple computations in MapReduce

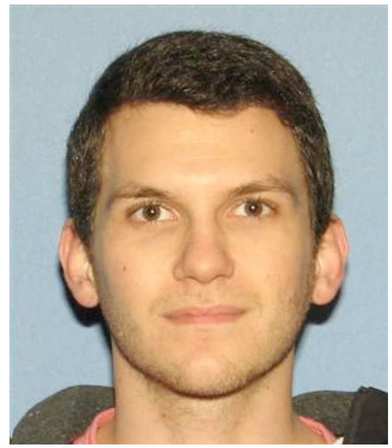
You will not be asked to write detailed Pig Latin code, but should have some basic understanding of how queries are implemented over MapReduce

That's it (almost)

But first…

Thanks to the folks who made it work!









One more thing…

Course evaluations

- Constructive feedback (positive we hope, but negative when called for) is what helps us get better
- Please fill out online by Sunday

And that's it…

Congratulations on a productive quarter!

Good luck on finals, last projects!!

See you Monday for the final exam!!