The Entity Relationship Model and Logical Database Design
- Ovals for columns, underline for primary key
- Rectangles for tables
- Diamonds for relations
- Fat connector lines for weak entity sets (one-to-many relations)
- Aggregation is a dashed box around multiple entities/relations

The Relational Model and Relational Algebra
- SELECT * FROM table WHERE y>3: σ y>3 (table)
- SELECT x FROM table: π x (table)
- Combined as: π x (σ y>3 (table))
- Join: |><|
- Except: -
- Simple join (cartesian product): X
- Union: U
- Intersect U down

Structured Query Language (SQL)
- *Read RM:RA and Index Structures*
- CREATE TABLE, UPDATE, INSERT INTO, SELECT, CREATE VIEW
- Never use SELECT *
- Primary key constraints
- Foreign key constraints, cascade on delete/update
- Nested selects
- Null values: IS NOT NULL, not != NULL.
- Triggers

Index Structures
- Data Entry for Indexes:
  - 1: {Key, actual data record} pairs
  - 2: {Key, record ID} pairs
  - 3: {Key, list of record IDs} pair
- Index Clustering
  - Clustered: Same order as data (1)
  - Unclustered: Data is random order (2,3)
- Indexes
  - Hash based: Buckets (non-unique, tree'ish, can be unique for partial keys), or rehash (unique). Great for equality checks, bad for ranges.
  - Tree based: Boundaries that lead to branches, and eventually to leaves (data entry lists). B+ trees (balanced) so that all paths are same length. Great for ranges, also good for equality.
- Composite keys: Indexes on multiple columns.
- ISAM: Tree structure never changes (static) so results can overflow on extra pages. Doesn't need locks for accessing the index.
- B+ Tree: Dynamic, so no overflow pages.

External Sorting
- Falls apart if the disk is heavily fragmented otherwise.
- When sorting happens: Ordering, index bulk-loading, duplicate elimination, joins.
- Merge sorts, blocked I/O (read in sequential blocks in 1 command, as opposed to X commands), double buffering.
- Using indexes: Clustered merging is fast, unclustered slow.

Evaluation of Relational Operators

Query Optimization

- Catalogs, tables which include system catalogs
  - Tables: name, filename, structure (heap, sorted), attributes (columns) with types, indexes, constraints (primary, foreign keys)
  - Indexex: name, structure (hash, B+ tree), search keys (columns)
  - Views: name, definition
  - Statistics: number of rows, size, cardinality of indexes, index size, index height, index range
- Common techniques: Indexing (good for conditions), Iteration (sequential scan, bad), Partitioning (use index to get smaller set, then iterate over that set)
- Conjunctive normal form (CNF): column OP value, where OP is any of the 6 comparison ops. Hash must match entire key, while trees can match partial keys.
  - Primary conjuncts: columns that qualify for limiting output by keys
- Selection: ”SELECT *”: pick the most selective access path (least amount of rows)
- Projection: ”SELECT columns”: same as Selection. ”SELECT DISTINCT”: partition using available keys, and discard duplicates.
- Join: ”SELECT c,a FROM ta, tb USING (b) WHERE ta.c=2”: index nested loop join
- Union, Intersect: like Projection
- Group By: use indexes, temporary aggregate counters
- Pipelining (piping) as opposed to temp tables (usually but not always good). Restrict before join, if that's good. Join re-ordering.

Concurrency Control

- ACID transactions: Atomicity, consistency, isolation, durablity
  - Atomic: Either all or nothing.
  - Consistent: The user must maintain consistency.
  - Isolation: A transaction is isolated from all other concurrent transactions.
  - Durability: Once reported completed, it's on disk.
- Interleaved transaction execution, serializability
  - Anomalies/conflicts:
    - Reading uncommitted data (WR)
    - Unrepeatable reads (RW)
    - Overwriting uncommitted data (WW)
  - Locking:
    - Strict Two-Phase Locking (Strict 2PL)
      - Shared, exclusive

Durability (Storage Stability and Crash Recovery)
Schema Refinement and Normal Forms
Database Tuning.