COSC386-Databases

Final Review

Table of Contents

Page 1: Notes/Defns
Page 2+: Examples

JJ McCauley

S'25

Lotta theory & stuff

Notes & Examples

Functional Dependency:

- There is a formal defn, but just think of it like:
  - ↳ If you know A, then you can determine B
  - ↳ A is the determinant, B is the dependent.
  - → This makes it super easy to understand. Essentially, each value of A should be associated w/ exactly one value of B for every instance of the relation.
  - ← ★Notecard (if space permits)
- "Superkey" = superset key
  - ↳ Not candidate key (not minimal)
- a "key" must be minimal
  - ↳ attributes "functionally determines" all other attributes
- Rules:
  - ↳ Armstrong's inference rules
    - **Reflexive** If $Y \subseteq X$, then $X \to Y$ ← This is also called "trivial"
    - **Augmentation** If $X \to Y$, then $XZ \to YZ$, where $XZ = X \cup Z$, & $YZ = Y \cup Z$
    - **Transitive** If $X \to Y$ and $Y \to Z$, then $X \to Z$
  - ↳ Additional Rules - Union
    - **Union** $a \to b$, $a \to c$; then $a \to bc$
    - **Decomposition** $a \to bc$; then, $a \to b$ and $a \to c$
    - **Pseudotransivity** $a \to b$, $cb \to r$; then, $ac \to r$

★Notecard (all rules)

- Attribute Closure Sets (x^+)
  - ↳ Given set of attributes X and set of FDs, the set of all attributes implied by X.

Decomposition (BCNF, 3NF):

- "Lossless" → If we can guarantee $R = S \bowtie T$
- Boyce-Codd Normal Form (BCNF)
  - ↳ Follows to primary conditions (if either one holds)
    1) $X \to Y$ is trivial ($Y \subseteq X$) ← Is the right side (Y) already contained in the left side (X)? Ex) $AB \to A$, $ABC \to BC$, etc

       OR

    2) X is a superkey for R

    ★Notecard
  - ↳ Limitations
    - Designed for Lossless join, dependency preservation, and Anomaly-free
    - Always excludes anomalies, May give up dependency preserving
- Third Normal Form (3NF)
  - ↳ Subset of BCNF w/ one additional rule
    1) $X \to Y$ is trivial
    2) X is a superkey for R

       AND

    3) Y is a proper subset of some key of R (BCNF is not always dependency preserving)

    ★Notecard
  - ↳ Always allow lossless join and dependency preserving
  - ↳ May allow some anomalies (update, insertion, deletion)

Transactions

↳ A sequence of actions that must either be executed as a whole, or non at all.

↳ Desired Properties of a Transaction (ACID) (requirements)
- → **Atomicity**: All steps of a transaction must fully complete or not at all (no partial executions)
- → **Consistency**: Transaction must preserve database rules/invariants
- → **Isolation**: Concurrent transactions must not interfere with each other (should be as if each happened one at a time)
- → **Durability**: Once a transaction commits, it's changes must persist, even after crash

↳ Transaction States
- → Active
- → Partially Committed
- → Failed
- → Aborted
- → Committed

↳ SQL Syntax

```
START TRANSACTION
  : Logic Here, multiple statements
COMMIT or ROLLBACK (=ABORT)
```

↳ Within a Transaction, we can say:

`SET TRANSACTION ISOLATION LEVEL X` where X =

This allows us to define what interactions are allowed by Transactions that execute around the same time.

X can be set to:

★Notecard (brief)

1) SERIALIZABLE
   - Strongest isolation level
   - Transactions behave as if they were executed one at a time
   - Safest but slowest
2) REPEATABLE READ
   - A transaction sees the same data when reading the same row multiple times
   - Allows Phantom Reads (if new rows appear as the result of other commands)
3) READ COMMITTED
   - A transaction only sees committed data from other transactions
   - Allows non-repeatable reads & Phantom reads, prevents Dirty Reads
   - May read different values if rereading same row
   - Usually the default
4) READ UNCOMMITTED
   - Lowest isolation; allows viewing of uncommitted changes from other transactions
   - Fastest, but very unsafe (inconsistent reads)

Dirty Read: Reading data modified but not yet committed

Non-repeatable Read: When the same row is read twice, and the values changed bc another Transaction updated it

Phantom Read: When a SELECT query returns different rows if ran multiple times, due to inserts/deletes by another transaction

Safety: How safe & consistent the level is

Level	Dirty Read	Non-repeatable Read	Phantom Read	Safety
Serializable	✗	✗	✗	✓✓✓
Repeatable Read	✗	✗	✓	✓✓
Read Committed	✗	✓	✓	✓
Read Uncommitted	✓	✓	✓	✗

↓ Isolation Level

★Notecard

Overarching Conceptual Review

Table 1: One-Page Refresher of Key Database Concepts for the Final Exam

#	Concept	1–2 Sentence Summary
1	Relational-Model Basics	A relation is a table-like set of tuples whose schema specifies attribute names and domains; data integrity relies on keys/superkeys and constraints (primary, foreign, domain).
2	E–R Diagrams	E–R diagrams model data as entity sets with attributes and relationships with cardinalities; they also capture weak entities and specialization/generalization.
3	E–R → Relational Mapping	Each strong entity becomes a table; relationship sets map to tables or foreign keys based on cardinality, and weak entities inherit key parts from their owners.
4	SQL Sublanguages	DDL creates/alter/drops schema objects, while DML retrieves or modifies data via `SELECT`, `INSERT`, `UPDATE`, and `DELETE`.
5	SQL Fundamentals	Core `SELECT` handles projection, selection, joins, ordering, pattern matching (`LIKE`), and set operations (`UNION`, `INTERSECT`, `EXCEPT`).

continued on next page

continued from previous page

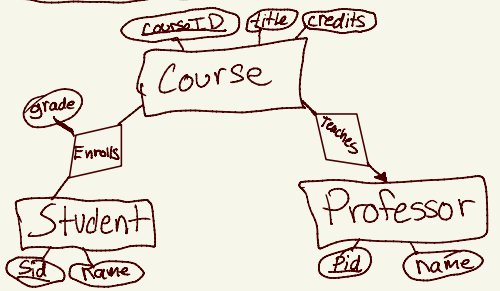
#	Concept	1–2 Sentence Summary
6	SQL Aggregation & Grouping	Aggregate functions (`MIN`, `MAX`, `AVG`, `SUM`, `COUNT`) work with `GROUP BY` to partition rows and `HAVING` to filter groups.
7	SQL Nested Queries	Subqueries in `WHERE`, `FROM`, or `SELECT` enable correlated tests (`EXISTS`, `IN`) and derived tables.
8	Relational Algebra	Operators such as selection (σ), projection (π), joins, and set operations ($\cup$, $\cap$, $-$) form the procedural foundation of SQL.
9	Basic PHP–SQL Interface	PHP connects to databases via PDO / mysqli, uses prepared statements and bound parameters to execute SQL securely and avoid injection.
10	Functional Dependencies	An FD $X \to Y$ holds when every value of X determines exactly one value of Y; Armstrong's axioms and attribute closure help infer keys.
11	Normalization (3NF, BCNF)	Decompose schemas so every non-key attribute depends only on candidate keys (3NF) or determinants are keys themselves (BCNF), eliminating update anomalies.

continued on next page

continued from previous page

#	**Concept**	**1–2 Sentence Summary**
12	Transactions & ACID	A transaction enforces Atomicity, Consistency, Isolation, and Durability; concurrency control and recovery mechanisms put these guarantees into practice.

ERD Practice (Brief)



TO
Schema

Professor(Pid, name)
Course(courseID, title, credits, foreignkey(profID references Professor(pid))
Student(SID, name)
EnrolledIn(foreignkey(SID references Student(SID), foreignkey(courseID references Course(courseID), grade)

(EZ)

Fk → foreignkey(attrname references Table(attrname))

Strong Relationships → Own Table (w/ foreign keys)
Weak Relations → Combine w/ Owner's primary key (ownerID, partialID)
1-to-1/1-to-Many → Add foreign key to one side, avoid new table
Relation has attrs? → New Table

Subclasses (ISA) → OO style (1 per table), ER style (shared PK), or NULLs