Introduction to Relational Databases, SQL, JDBC

CSCE 156 – Computer Science II

1. Introduction
    a. Database concepts
        i. Persistence of data across systems, sessions, etc.
    b. Your MySQL database on CSE
        - Database name: same as login
        - Change Password: ponca.unl.edu
        - Command line usage: mysql –u user –p user
        - Command line redirection, mysqldump, etc.
2. Relational Databases
    a. Data (flat file problems)
    EXAMPLE
        - Repetition of data
        - Incomplete data
        - Integrity of data
        - Organizational problems (to determine simple things like counts requires processing entire file)
        - Updating information can be difficult (must enumerate all possible changes, entire file is processed)
        - Formatting issues (delimiters)
        - Concurrency issues
    b. Key aspects
        - RDBS (Relational* Database Systems) store data in *tables*
        - Tables have a unique name and description of types (integer, string) of data
        - Each column stores a single piece of data (field)
        - Each row represents a record/object
        - Each row may have a unique *primary key* (automatically incremented, unique identifier—NUID; combination of fields—Geo data)
        - Rows in different tables are related through *foreign keys*
        - Order of rows/columns meaningless
        - Constraints (nullity, bounds, enforced formatting, etc.)
        - ACID
            - Atomicity – modifications must be all or nothing (atomic operation, not divisible or decomposable)

        - Consistency – transaction will retain state of consistency (constraints, cascades, triggers)
        - Isolation – No transaction interferes with another
        - Durability – Once committed, a transaction remains so (protected against power loss/crash)
    - Examples
        - MS Access (hahaha)
        - MySQL (GNU GPL, owned by Oracle)
        - PostgreSQL (FOSS)
        - Informix (IBM)
        - DB2 (IBM)
        - SQLServer
        - Oracle Database
- c. Advantages
    - Data is structured instead of "just there"—better organization
    - Duplication is minimized (with proper normalization)
    - Updating information is easier
    - Organization of data allows easy access
    - Organization allows aggregation and more complex information
    - Data integrity can be enforced (data types and user defined constraints)
    - Faster
    - Scalable
    - Security
    - Portability
    - Concurrency
- d. Structured Query Language
    - Common language/interface to most databases
    - Developed by Chamberlin & Boyce at IBM, 1974
    - Implementations may violate standard, portabilitiy problems
    - Comments: #--
    - Create, manage tables (CREATE ALTER DROP
    - CRUD – Create, Retrieve, Update, Delete
    - Transactions (MySQL: begin transaction; rollback; or commit;)
- e. Misc Issues
    - Views
    - Triggers
    - Stored Procedures

3. Tables
    a. Creating Tables
        i. Syntax:
        CREATE TABLE TableName (

```
    field_name fieldType [options],
    PRIMARY KEY (keys)
);
```

ii. Options
- NOT NULL
- AUTO_INCREMENT
- DEFAULT (value)

iii. MySQL helpful commands (WARNING)
- USE *database*;
- SHOW TABLES;
- DESCRIBE *table*;

b. Column Data Types
- VARCHAR(n) (also CHAR, NCHAR, NVCHAR)
- INTEGER (INT, SMALL INT)
- FLOAT (FLOAT, REAL, DOUBLE PRECISION)
- DECIMAL(n,m) (NUMERIC(n,m))
- Date/Time functions: rarely portable, for MySQL functions: http://dev.mysql.com/doc/refman/5.0/en/date-and-time-functions.html

c. Primary Keys
- Need a way to distinguish records
- At most one primary key per table
- Must uniquely identify all possible records (not just those that exist)
- No two rows can have the same primary key value
- PKs can be one or more columns—combination of values determines key
- Should not use/allow NULL values
- Can/should* be automatically generated (let the database handle it)

d. Keys
- Can have multiple keys
- May be a combination of columns
- NULLs are allowed—may result in multiple rows
- Uniqueness is enforced (updates, inserts may fail)
- May be declared non-unique in which case it serves as an index (for database optimization)

e. Foreign Keys
- Relations can be made between tables using FKs
- A FK is a column that references a key (PK or K) in another table
- Inserts cannot occur if the referenced record does not exist (NULL issue)
- Usually establishes a one to many relationship
- Table with FK (referencing table) references table with PK (referenced table)

- Cascades (evil): deleting rows in the referenced table cascade to the referencing records (which are deleted)

4. Manipulating Data
    a. Inserting Data
    b. Querying Data
        i. SELECT
            1. Good Practice: be intentional (enumerate fields, us AS)
        ii. Clauses
            1. WHERE
                a. wildcards
            2. ORDER BY
            3. GROUP BY
            4. HAVING
        iii. JOINs
        iv. Temporary Tables
        v. Nested Queries
    c. Deleting data
        i. CAREFUL: always use WHERE clause!
5. Designing a Database: a class roster
    a. Identify entities:
        i. Student
            1. NUID: primary key? External, some begin with zero, all 8 digits (additional key)
            2. Name (first/last? Middle?): constraints (not blank, case sensitive)?
            3. Email: Break out into another table, RFC 2821
        ii. Course
            1. Id?
            2. Title
            3. Description
        iii. Enrollment
            1. Many-to-many: foreign keys
            2. Semester (representation? 20111 versus 1111, date, etc?)
6. JDBC – Java Database Connectivity API
    a. API Overview
    b. Establishing a connection
    c. Making a query
    d. Prepared Statements
    e. Handling Exceptions, cleaning up
7. Good practice (design) (TODO: move this down)
    - Use standard SQL!
    - Use consistent naming conventions

- Use keys to enforce referential integrity
- The AUTO_INCREMENT problem (with relations)
- Use constraints to enforce data integrity
- Normalization
- Use transactions!

8. References
    - MySQL 5.1 Reference Manual (http://dev.mysql.com/doc/refman/5.1/en/index.html)
    - MySQL Community Server (http://www.mysql.com/downloads/)
    - MySQL Workbench – a MySQL GUI (http://wb.mysql.com/)
    - Connector/J (MySQL JDBC connector): http://www.mysql.com/downloads/connector/j/
    -