Databases (CSCI2141)
Covers SQL to :

Lecture Dates: February 28th

Midterm Answers

1 T	2 T	3 T	4 T	5 T	6 F	7 T	8 T	9 T	10 T
11 T	12 T	13 T	14 T	15 T					

Fill in the blanks will not be on final as they were on midterm.

16 Data independence
17 Strong
18 Equijoin
19 Total
20 Primary
21 Candidate
22 Repeating (talked about in 1NF)
23 Existence-dependent
24 Overlapping
25 BCNF
26 Partial dependency
27 Flags
28 Attribute
29 DIFFERENCE
30 Normal

31 Metadata	32 Data Inconsistency	33 Relationship (?)	34 ER 35 Synonym
36 Integrity	37 D	38 Simple	39 Need not physically...
40 Subtype D	41 Generalization	42 Single Ab.	43 1NF
44 Prime	45 Tr. Depen.		

46 Entity is existence-dependent. Entity has a primary key, partially/totally derived from parent entity in rship.

Weak entity in a weak relationship? No. Two conditions are mutually exclusive.

47 Tables share same number of cols. Compatible data types.

48 Homonymns.

49 Query.

50 Unique values, nonitelligent, not change over time, preferably numeric, composed of single attr,

security compliant, etc.

51 See class notes.
52 Crow's Foot ERD

53 All attributes of CD include Prod_... (Inheritance)
No (PRODUCT associated with every instance in CD) – total completeness.
No (Subtypes can only exist within context of a supertype)

54 Primary Key: ISBN,Author_Num

Partial Dependencies:	ISBN -> BookTitle, Publisher, Edition
	Author_Num -> LastName
Transitive Dependency:	BookTitle -> Publisher

2NF? No.
3NF? No.

Introduction to SQL

MySQL Accounts: created on Bluenose. Passwords initially set to your Banner ID.
Tutorial: Lab 3 – Friday March 2nd (11:30 to 1:00). Bring laptop with you.

- Project guidelines handed out (individual). Due: April 5th, 2012 (2:35PM).
  - Written report.
  - Not submitting entire database.
  - Layout of report up to us. Spelling, etc. Matter.
  - Different relationships, attributes.
  - Do not have to use advanced features (e.g. Weak entities, specialization hierarchies).
- SQL Handout given out. May not cover entire document.
- SQL is a Data Definition Language – create objects, define access rights.
  - Defining database objects.
  - Tables, database itself, etc.
- It is a Data Manipulation Language – insert, update, delete, retrieve data within tables.
  - Manipulating data, tables.
  - Inserting, changing, select, etc.
  - Roll-backs to an earlier form of a database can only work for data manipulating commands (see later).
- It is a Non-Procedural Language – specifies what must be done (not how it is done). Does not need to know data storage format, how actually executed.
- Multiple SQL "dialects"; minor differences. Moving from one RDBMS to another requires some modifications.
- Popular RDBMSs – Oracle, Microsoft SQL Server, MySQL (free), DB2 (IBM), Microsoft Access
- SQL queries can be questions, actions.
- Creating a Database requires creating the database structure.
  - Authentication process will differ from one RDBMS to another.

- Creation of physical files holding the database.

- Schema: Group of related DB objects. Usually belong to a singler user, application.
    - First level of security.
    - Each user sees only *their* tables.
- Data Types: Great deal of them. (Chapter 8). Basic types: NUMBER, INTEGER, CHAR, VARCHAR, DATE. Within each of them, a lot of variety (varies between systems).
- Data Type Selection is dictated by:
    - Nature of data.
    - Intended use.
- Be able to defend your choice.
- Character data is processed quicker in queries; not performing mathematical procedures. (Phone number, etc.). Example: American ZIP codes (can have a leading 0).
- When choosing a data type, consider expected use of sorting, data-retrieval purposes.
- SQL Example:
    - CREATE DATABASE database_name;
    - USE database_name; changes current "working database".
    - DROP DATABASE database_name;
- Creating Table Structures
    - Enclose list fo attributes in one set of paranthesis. Use one line per column (attribute) definition. Stylistic convention.
    - Separate table elements with comma. No comma after last element.
    - End each statement with a semicolon.
- SHOW DATABASES; SHOW TABLES; shows all tables.
    - SHOW COLUMNS FROM table_name;
    - DESCRIBE table_name; does the same thing.
    - DESC table_name; does the same thing. Does not show data. Metadata.
- Deleting a Table
    - DROP TABLE table_name;
    - Deletes data, definition. Can have comma-separated list.
- Foreign Keys
    - FOREIGN KEY(V_CODE) REFERENCES VENDOR(V_CODE – *primary key for that table name*) ON UPDATE CASCADE
    - Update Cascade: Constraint. Ensures referential integrity. Cascades all changes.
- Constraints
    - Example: NOT NULL, UNIQUE
    - Can be defined at column definition (column constraint) – applies to just one column.
    - Table constraint (CONSTRAINT keyword) – applies to many columns.
    - ON UPDATE CASCADE ensures changes to foreign key are applied to all references. Support varies from one RDBMS to another.
    - ON DELETE CASCADE recommended for weak entities. Ensures deletion of a row in strong entity automatically triggers deletion of rows in dependent.
    - DEFAULT constraint makes the value default to something.
    - CHECK constraint checks to see if values match a condition (example, in a list of certain values). Checks when added, modified.
- Default behavior of DECIMAL datatype: 10 before decimal, 0 after.

- SQL Indexes can be created for any selected attribute.
  - CREATE [UNIQUE] INDEX index_name ON table_name(column1,[column2]);
  - See slides.
  - Common to create an index on any attribute used as:
    - a search key
    - a conditional expression
    - a way to list rows in specific order.
- Composite indexes are often used to prevent data duplication.
- By default, indeces produce results listed in ascending order. Can do it in descending order: ON table_name(column1 DESC)
- Adding table rows.
  - Row contentsentered between paranthesis.
  - NULL used for unknown values.
  - See slide.
- Adding rows with optional attributes: can only list attributes with values and attribute names listed first.
  - See slide.
  - Listing of attribute names for what you are entering.
- Update is used to modify table data. UPDATE is a set-oriented operator. Will update every row unless you put a constraint. WHERE condition optional. See slide.
- Inserting table rows with SELECT.
  - Values returned by SELECT must match attributes, data tpes of table in INSERT statement.
  - Essentially, rows can be added using another table as source of data.
  - SELECT is a subquery in this case – nested query or inner query.
  - Queries can be nested many levels deep.
  - Queries executed starting with innermost and working out.
- DELETE is another set-oriented command.
- Changes to table contents not saved on disk until database is closed, program is closed, or COMMIT command is used. COMMIT;
- ROLLBACK undoes any data manipulation changes made since last COMMIT. ROLLBACK;
  - Does not undo data definition commands.
  - For example, will not take it back to original condition sometimes.
- Seeing all the contents of a table: SELECT * FROM table_name;
- Correction on Handout: Remove NOT NULL from CUS_AREACODE in CUSTOMER table.
  - Using NOT NULL negates the point of DEFAULT.
  - MySQL does not support verification used by CHECK constraints in other RDBMSs.
  - Can put in code, but does not actually do the verifications.
- SELECT queries: can be fine-tuned by adding restrictions to search criteria – Conditional, Arithmetic, Logical, Special operators.
  - Conditional Expressions: WHERE clause.
    - Reminder: String Comparisons
    - Made from Left to Right. See slide.
    - Aliases can be used for any column in a SELECT query. See slide.
  - Arithmetic: See slide.
  - Logical: Using WHERE. Includes OR, AND, etc.

    - Special: Used in conjunction with WHERE clause. Example: BETWEEN, IS NULL, LIKE, IN, EXISTS. Cannot use equality operator to test for NULL (not a value). Wildcards (% _).
- ALTER TABLE command. Add, modify, delete columns in existing tables. Can also be used to add, remove a constraint.
    - ADD (one or more columns). Do not include NOT NULL clause.
    - MODIFY changes column characteristics.
        - Depending on RDBMS: column data types can only be changed if column is empty.
        - OR column widths can only be increased (and not decreased).
    - DROP: deletes a column.
        - Depending on RDBMS: Attributes that are foreign keys cannot be deleted.
        - OR An attribute that is only attribute may not be able to be deleted.
        - OR Column containin data may not be able to be deleted.
- UPDATE command updates data in existing rows. Arithmetic operators can be used.
- Copying Parts of Tables: Contents of selected columns can be copied so data does not need be re-entered manually into new tables. See slide.
    - Using Insert.
    - Shorter way to do it – does not apply primary or foreign keys.
- See SQL exercise.
- Can e-mail her at: JANIE.MASON@HOTMAIL.COM
- Adding Primary and Foreign Key Designations
    - When table is copied, integrity rules do not copy.
    - Primary, foreign keys are manually defined on new tables. Use ALTER TABLE command.
- Ordering a List
    - ORDER BY clause is useful when listing order is important.
    - Default order is ascending.
    - Multi-level ordered sequence is known as a cascading order sequence; created by listing several attributes, separated by commas, after ORDER BY clause.
- SELECT columnlist FROM tablelist [WHERE conditionlist] [ORDER BY columnlist [ASC | DESC]];
- DISTINCT clause products list including only values different from one another. Nulls placed at top or bottom depending on RDBMS.
- Aggregate Functions incl. COUNT, MIN, MAX, SUM, AVG.
    - MIN, MAX, AVG can only be used in SELECT.
    - COUNT can be used on its own.
- GROUP BY can be used to create frequency distributions. Used in conjunction with one of the aggregate functions.
    - See slide.
    - HAVING clause (similar to WHERE in SELECT).
    - Even though have alias, cannot use it in HAVING or ORDER BY clause. May depend on system.
- CREATE VIEW – virtual table based on SELECT query.
    - Name of view can be used anywhere you can use a table.
    - Are dynamically updated. Recreated each time invoked.
    - May be used as basis for reports.
- Joining Database Tables

  - Usually an equality comparison between foreign and primary key of related tbales.
  - Join tables by listing tables in FROM clause of SELECT statement.
- Recursive Joins
- Left Outer Join (LEFT JOIN)
- Right Outer Join (RIGHT JOIN)
- Chapter 8 Overview
  - Relational set operators combine output of two queries to generate new relations.
  - Operations that join tables are classified as inner, outer joins.
  - Natural join returns all rows with matching values – eliminates duplicate columns.
- Relational Set Operators
  - UNION (UNION DISTINCT – different than UNION ALL which retains all duplicates)
  - INTERSECT
  - MINUS
  - Relations must be union-compatible.
- IN subqueries.
- INTERSECT (see slides).
  - Alternative Syntax – if cannot use INTERSECT operator.
  - Using SELECT ... FROM ... WHERE ... AND ... (see slide). Nested query.
- MINUS (SET DIFFERENCE) – sometimes called EXCEPT.
  - Not supported in MySQL.
  - Alternative Syntax – another nested query (see slide).
- CROSS JOIN – performs a relational product (Cartesian product) of two tables.
  - See slides.
  - SELECT ... FROM ... CROSS JOIN ... ;
- NATURAL JOIN – does not require all table qualifiers. Eliminates duplicate columns.
  - For example, do not need to use aliases, dot operators.
  - See steps on slide.
  - Can be done on multiple tables at any one time.
  - Is used when tables share one or more common attributes with common names.
- JOIN USING clause – using common attribute.
- JOIN ON clause – used when tables have no common attributes. Table qualifiers must be used.
- Outer Joins – returns matching rows AND unmatched rows, unmatched values will be null. Left, right, full.
  - FULL OUTER JOIN – returns rows matching join condition, unmatched values on either side.
  - See slides.
- ***Subquery Queries***
  - Subquery: Query inside a query (nested query); normally inside paranthesis.
  - First query is the outer query; inside query is the inner query.
  - *Example*: WHERE Subquery. Most common type of subquery.
    - Must return a single value; return value must be a comparable data type.
    - See slide.
  - *Also*: HAVING Subquery. Restrict output by GROUP BY query.
  - *Also*: IN Subquery. See slides.
  - Multirow Subquery Operators: ANY and ALL

        - IN operator uses equality comparison.
        - ANY and ALL operators allow comparison of single value with a list of values using inequality. Using “ = ANY” is equivalent to using IN operator.
    - *Also*: FROM Subquery.
    - Attribute List Subquery
        - When using SELECT statement, columns can be attributes from existing tables, computed, aggregate function results.
        - The attribute list can also contain a subquery (***inline query***). Must return a single value; will be same in every row in output.
        - Column Aliases cannot be used in computations in the attribute list when alias is defined in the same attribute list.
    - Correlated Subquery
        - Similar to a for loop.
        - Once for each row in outer query. Inner query related to outer query (correlated).
        - Passing outer row to inner row at every iteration.
        - See slides.
- SQL Functions
    - Tend to change from DB to DB.
        - Know that they are there; know in general what looking for.
        - Names, syntax will differ.
    - Think programming languages; use numerical, data, string input.
    - Can appear anywhere in an SQL statement.
    - Date, Time Functions; can do date arithmetic, parsing.
    - Numeric Functions; algebraic, trigonometric, logarithmic, etc. Do not confuse with aggregate functions (operate over sets vs. single row). Take one numeric parameter and return one value.
    - String Functions; string manipulation functions like toUpper, etc.
- Assignment 4 passed back.
    - T/F: TFTTTFTFFT
    - Fill: 11 Data Redundancy, 12 3NF, 13 one, 14 NULL, 15 DROP TABLE, 16 HAVING, 17 index, 18 EXISTS, 19 ROLLBACK, 20 UPDATE
    - Short: 21 Diagram, 22 Done in-class, 23 / 24 / 25 / ...
- Procedural SQL
    - SQL does not support conditional execution of procedures typically supported by programming languages such as: if <condition> then <...> else <...>;
    - Nor does it support looping operations like WHILE.
    - SQL-99 standard defined use of persistent stored modules (PSM). Blocks of code containing standard SQL statements, procedural extensions. Stored, executed at DBMS server.
    - Procedural code is executed as a unit. Invoked by end user. Can create:
        - Anonymous PL/SQL blocks (not given a name).
        - Triggers
            - Automatically invoked by a data manipulation event (before, after insertion, update, deletion).
            - For instance, everytime an insertion, there a procedure run to do certain updates.

        - They can:
            - Enforce constraints.
            - Provide warnings.
            - Update table values.
            - Inserting records.
            - Call other stored procedures.
        - Can be used to:
            - Automatically generate derived column values.
            - Enforcement of business, security constraints.
            - Create replica tabels for backup purposes.
        - Syntax – see slide.
            - Name typically starts with TRG_
            - If "FOR EACH ROW" not there, trigger run only once.
        - :NEW and :OLD Attribute References – two copies of every row being changed by INSERT, UPDATE, DELETE. First copy has "old" values before changes, second copy has "new" changed values. Can only be used in a PL/SQL trigger action.
    - Stored procedures.
    - PL/SQL functions.
  - At the SQL prompt, can type SET SERVEROUTPUT ON. Allows you to add output lines. Use the DBMS_OUTPUT.PUT_LINE function.
- MySQL & PHP Together
  - Embedding PHP into an HTML file (<? php ?>)
  - Variables in PHP often start with a $ sign.
  - Use the mysql_connect() function, mysql_query($query), mysql_close();
  - Can use HTML Forms to send data to PHP and in tandem, insert it into a database.
- From Handout, "Create a trigger named..."
  - Answer on handout.

```
CREATE OR REPLACE TRIGGER TRG_MEM_BALANCE
AFTER UPDATE OF DETAIL_DUEDATE, DETAIL_RETURNDATE
     ON DETAIL RENTAL
FOR EACH ROW
DECLARE
     PRIOR_LATEFEE NUMBER;
     CURRENT_LATEFEE NUMBER;
     UPDATE_AMOUNT NUMBER;
     RENTAL_MEMBER RENTAL.MEM_NUM%TYPE;
BEGIN
     PRIOR_LATEFEE:=:OLD.DETAIL_DAYSLATE *
          :OLD.DETAIL_DAILYLATEFEE;
     IF PRIOR_LATEFEE IS NULL THEN
          PRIOR_LATEFEE:=0;
     END IF;
     CURRENT_LATEFEE:=:NEW.DETAIL_DAYSLATE *
          :NEW.DETAIL_DAILYLATEFEE;
     IF CURRENT_LATEFEE IS NULL THEN
```

```
          CURRENT_LATEFEE:=0;
     END IF;
     UPDATE_AMOUNT:=CURRENT_LATEFEE – PRIOR_LATEFEE;
     IF UPDATE_AMOUNT > 0 THEN
          SELECT MEM_NUM INTO RENTAL_MEMBER
               FROM RENTAL WHERE RENT_NUM = :NEW.RENT_NUM;
          UPDATE MEMBERSHIP
               SET MEM_BALANCE = MEM_BALANCE
                    + UPDATE_AMOUNT
               WHERE MEM_NUM = RENTAL_MEMBER;
     END IF;
END;
/
```

- Assignment #5 Solutions
    - Multple Choice: 1 A, 2 A/B/C/D – in practice, 3 A, 4 D, 5 C, 6 B, 7 A, ..., 10 A, ..., 13 A, 14 C, ...
    - Short Answer: 16 – from slide, 17, ..., 23
- Conditional Trigger Actions
- Trigger Variables
- Stored Procedures