**CSCI 4333.2 Classroom Notes and Demonstrations**

8/27/2025

**1.2 The Relational Model: an introduction**

* The basic relational data model in layman terms:
  + A database is composed of a collection of *tables* (relations). Structured db.
  + A table contains many *rows* (tuples) (one row for data of one concept/object/…) and *columns* (attributes/fields): properties.
  + Each row contains many *column values*.
  + Every row of a table has the same set of columns.
  + Values of the same column have the same data *type*.
  + Keys are sets of columns/attribute.
  + A *candidate key* of a table is a *minimal unique identifier* of a row in the table.
  + A *primary key* is a selected candidate key (for storing the table).

A screenshot of a computer

AI-generated content may be incorrect.

SQL to create the relation schema/structure of student:

|  |
| --- |
| CREATE TABLE IF NOT EXISTS Student (  stuId INT NOT NULL,  fname VARCHAR(30) NOT NULL,  lname VARCHAR(30) NOT NULL,  major CHAR(4) NULL,  minor CHAR(4) NULL,  -- ach: accumulated credit hours, including transferred credits.  ach INTEGER(3) UNSIGNED NULL DEFAULT 0,  advisor INT NULL,  CONSTRAINT Student\_stuId\_pk PRIMARY KEY(stuId),  -- an artificial example of a CHECK constraint.  CONSTRAINT Student\_ach\_cc CHECK ((ach>=0) AND (ach < 250)),  CONSTRAINT Student\_major\_fk FOREIGN KEY (major)  REFERENCES Department(deptCode) ON DELETE CASCADE,  CONSTRAINT Student\_minor\_fk FOREIGN KEY (minor)  REFERENCES Department(deptCode) ON DELETE CASCADE,  CONSTRAINT Student\_advisor\_fk FOREIGN KEY (advisor)  REFERENCES Faculty(facId)  ); |

A screenshot of a computer

AI-generated content may be incorrect.

stuId ({stud}) is a CK/PK of the table student.  
stuId is not a CK of the table enroll.

A screenshot of a computer

AI-generated content may be incorrect.

|  |
| --- |
| CREATE TABLE IF NOT EXISTS Enroll(  stuId INT NOT NULL,  classId INT NOT NULL,  grade VARCHAR(2) NULL,  n\_alerts INT NULL,  CONSTRAINT Enroll\_classId\_stuId\_pk PRIMARY KEY (classId, stuId),  CONSTRAINT Enroll\_classNumber\_fk FOREIGN KEY (classId)  REFERENCES Class(classId) ON DELETE CASCADE,  CONSTRAINT Enroll\_stuId\_fk FOREIGN KEY (stuId)  REFERENCES Student (stuId) ON DELETE CASCADE,  CONSTRAINT Enroll\_grade\_fk FOREIGN KEY (grade)  REFERENCES Grade (grade) ON DELETE CASCADE  ); |

* + *Alternative/secondary keys* are candidate keys not selected as the primary key.
  + A *foreign key* of a relation references a primary key of another relation (known as the parent or referenced table).

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

|  |
| --- |
| CREATE TABLE IF NOT EXISTS Student (  stuId INT NOT NULL,  fname VARCHAR(30) NOT NULL,  lname VARCHAR(30) NOT NULL,  major CHAR(4) NULL,  minor CHAR(4) NULL,  -- ach: accumulated credit hours, including transferred credits.  ach INTEGER(3) UNSIGNED NULL DEFAULT 0,  advisor INT NULL,  CONSTRAINT Student\_stuId\_pk PRIMARY KEY(stuId),  -- an artificial example of a CHECK constraint.  CONSTRAINT Student\_ach\_cc CHECK ((ach>=0) AND (ach < 250)),  CONSTRAINT Student\_major\_fk FOREIGN KEY (major)  REFERENCES Department(deptCode) ON DELETE CASCADE,  CONSTRAINT Student\_minor\_fk FOREIGN KEY (minor)  REFERENCES Department(deptCode) ON DELETE CASCADE,  CONSTRAINT Student\_advisor\_fk FOREIGN KEY (advisor)  REFERENCES Faculty(facId)  ); |

A black background with white text

AI-generated content may be incorrect.

More theoretically:

* The (theoretical) relational model is based on the concept of a relation.
* It is a *set-theoretic* model: the definitions are based on mathematical sets.
* If you are not familar with set theory, read about it. This is a basic, short, good, and good-enough introduction: <https://www.ucl.ac.uk/~ucahmto/0005_2021/Ch2.S1.html> (note that in the set builder form, the author used ":" to represent "such that". We usually use "|" instead.)
* Note that practical DBMS do not implement the pure relational model.
* In the theoretical relational model:
  1. An *attribute* (*column/field*) is a name.
  2. A *domain* is a *set* of values an attribute can take.
     1. It is the set of values of the*data type* of the attribute.
     2. The value of an attribute should be *atomic* (cannot be divided into smaller components with individual meanings):
        1. If all attributes of a relation are atomic, the relation is said to be in *First Normal Form*.
     3. *Null* may or may not be an acceptable value for an attribute. It depends on problem requirements.
  3. A *relation schema*, R, is a *set* of attributes A1, A2,…,An with their domains D1, D2,…, Dn.
  4. A *tuple* (*row*) is a *set* of *mapping* of a *set* of attributes to a *set* of values: Ai -> di where di ∈ Di, for i = 1 to n (∈: belongs to)
  5. A *relation* (*instance*) is a set of tuples.
  6. The *degree* (or *arity*) of a relation is the number of attributes in its schema.
* Some advantages of the *relational model* and relational DBMS (as compared to other databases):
  1. Strong mathematical foundation
  2. Simple
  3. Strong design theory
  4. Strong support of data integrity and consistency
  5. Strong support of transactions
  6. Strong industrial support and community
  7. High popularity
* Some disadvantages of the relational model and relational DBMS:
  1. The data model may not match the problem requirements well.
  2. Impedance mismatch with object-oriented models.
  3. Do not scale well.
  4. Structured data may be too restrictive for specific problems.

**A First Look at SQL**

by K. Yue

**1. Introduction**

* The basic relational data model in layman and more theoretical terms can be seen again here: [RelationalModel.html](https://dcm.uhcl.edu/yue/courses/joinDB/Fall2025/notes/Intro/RelationalModel.html)

**2. Very Simple SQL**

Consider the [toyu](https://dcm.uhcl.edu/yue/courses/joinDB/Fall2025/notes/toyu/toyu.html) database.

* A simple SELECT statement in *SQL* an be used to retrieve data from a table.

***Example:*** in MySQL, executing:

SELECT \* FROM grade;  
SELECT \* FROM school;  
SELECT \* FROM department;  
SELECT \* FROM faculty;  
SELECT \* FROM course;  
SELECT \* FROM class;  
SELECT \* FROM student;  
SELECT \* FROM enroll;

Result: (MariaDB is compatible to MySQL).

MariaDB [toyu]> SELECT \* FROM grade;  
+-------+------------+  
| grade | gradePoint |  
+-------+------------+  
| A     |     4.0000 |  
| A-    |     3.6667 |  
| B     |     3.0000 |  
| B+    |     3.3333 |  
| B-    |     2.6667 |  
| C     |     2.0000 |  
| C+    |     2.3333 |  
| C-    |     1.6667 |  
| D     |     1.0000 |  
| D+    |     1.3333 |  
| D-    |     0.6667 |  
| F     |     0.0000 |  
| IP    |       NULL |  
| P     |       NULL |  
| WX    |       NULL |  
+-------+------------+  
15 rows in set (0.004 sec)  
  
MariaDB [toyu]> SELECT \* FROM school;  
+------------+-------------------------------+  
| schoolCode | schoolName                    |  
+------------+-------------------------------+  
| BUS        | Business                      |  
| EDU        | Education                     |  
| HSH        | Human Sciences and Humanities |  
| CSE        | Science and Engineering       |  
+------------+-------------------------------+  
4 rows in set (0.003 sec)  
  
MariaDB [toyu]> SELECT \* FROM department;  
+----------+------------------------------+------------+----------+  
| deptCode | deptName                     | schoolCode | numStaff |  
+----------+------------------------------+------------+----------+  
| ACCT     | Accounting                   | BUS        |       10 |  
| ARTS     | Arts                         | HSH        |        5 |  
| CINF     | Computer Information Systems | CSE        |        5 |  
| CSCI     | Computer Science             | CSE        |       12 |  
| ENGL     | English                      | HSH        |       12 |  
| ITEC     | Information Technology       | CSE        |        4 |  
| MATH     | Mathematics                  | CSE        |        7 |  
+----------+------------------------------+------------+----------+  
7 rows in set (0.000 sec)  
  
MariaDB [toyu]> SELECT \* FROM faculty;  
+-------+----------+----------+----------+---------------------+  
| facId | fname    | lname    | deptCode | rank                |  
+-------+----------+----------+----------+---------------------+  
|  1011 | Paul     | Smith    | CSCI     | Professor           |  
|  1012 | Mary     | Tran     | CSCI     | Associate Professor |  
|  1013 | David    | Love     | CSCI     | NULL                |  
|  1014 | Sharon   | Mannes   | CSCI     | Assistant Professor |  
|  1015 | Daniel   | Kim      | CINF     | Professor           |  
|  1016 | Andrew   | Byre     | CINF     | Associate Professor |  
|  1017 | Deborah  | Gump     | ITEC     | Professor           |  
|  1018 | Art      | Allister | ARTS     | Assistant Professor |  
|  1019 | Benjamin | Yu       | ITEC     | Lecturer            |  
|  1020 | Katrina  | Bajaj    | ENGL     | Lecturer            |  
|  1021 | Jorginlo | Neymar   | ACCT     | Assistant Professor |  
+-------+----------+----------+----------+---------------------+  
11 rows in set (0.001 sec)  
  
MariaDB [toyu]> SELECT \* FROM course;  
+----------+--------+--------+-------------------------------------+---------+  
| courseId | rubric | number | title                               | credits |  
+----------+--------+--------+-------------------------------------+---------+  
|     2000 | CSCI   | 3333   | Data Structures                     |       3 |  
|     2001 | CSCI   | 4333   | Design of Database Systems          |       3 |  
|     2002 | CSCI   | 5333   | DBMS                                |       3 |  
|     2020 | CINF   | 3321   | Introduction to Information Systems |       3 |  
|     2021 | CINF   | 4320   | Web Application Development         |       3 |  
|     2040 | ITEC   | 3335   | Database Development                |       3 |  
|     2041 | ITEC   | 3312   | Introduction to Scripting           |       3 |  
|     2060 | ENGL   | 1410   | English I                           |       4 |  
|     2061 | ENGL   | 1311   | English II                          |       3 |  
|     2080 | ARTS   | 3311   | Hindu Arts                          |       3 |  
|     2090 | ACCT   | 3333   | Managerial Accounting               |       3 |  
+----------+--------+--------+-------------------------------------+---------+  
11 rows in set (0.000 sec)  
  
MariaDB [toyu]> SELECT \* FROM class;  
+---------+----------+----------+------+-------+------+  
| classId | courseId | semester | year | facId | room |  
+---------+----------+----------+------+-------+------+  
|   10000 |     2000 | Fall     | 2019 |  1011 | D241 |  
|   10001 |     2001 | Fall     | 2019 |  1011 | D242 |  
|   10002 |     2002 | Fall     | 2019 |  1012 | D136 |  
|   10003 |     2020 | Fall     | 2019 |  1014 | D241 |  
|   10004 |     2021 | Fall     | 2019 |  1014 | D241 |  
|   10005 |     2040 | Fall     | 2019 |  1015 | D237 |  
|   10006 |     2041 | Fall     | 2019 |  1019 | D217 |  
|   10007 |     2060 | Fall     | 2019 |  1020 | B101 |  
|   10008 |     2080 | Fall     | 2019 |  1018 | D241 |  
|   11000 |     2000 | Spring   | 2020 |  1011 | D241 |  
|   11001 |     2001 | Spring   | 2020 |  1012 | D242 |  
|   11002 |     2002 | Spring   | 2020 |  1013 | D136 |  
|   11003 |     2020 | Spring   | 2020 |  1016 | D217 |  
|   11004 |     2061 | Spring   | 2020 |  1018 | B101 |  
+---------+----------+----------+------+-------+------+  
14 rows in set (0.001 sec)  
  
MariaDB [toyu]> SELECT \* FROM student;  
+--------+-----------+----------+-------+-------+------+---------+  
| stuId  | fname     | lname    | major | minor | ach  | advisor |  
+--------+-----------+----------+-------+-------+------+---------+  
| 100000 | Tony      | Hawk     | CSCI  | CINF  |   40 |    1011 |  
| 100001 | Mary      | Hawk     | CSCI  | CINF  |   35 |    1011 |  
| 100002 | David     | Hawk     | CSCI  | ITEC  |   66 |    1012 |  
| 100003 | Catherine | Lim      | ITEC  | CINF  |   20 |    NULL |  
| 100004 | Larry     | Johnson  | ITEC  | NULL  |   66 |    1017 |  
| 100005 | Linda     | Johnson  | CINF  | ENGL  |   13 |    1015 |  
| 100006 | Lillian   | Johnson  | CINF  | ITEC  |   18 |    1016 |  
| 100007 | Ben       | Zico     | NULL  | NULL  |   16 |    NULL |  
| 100008 | Bill      | Ching    | ARTS  | NULL  |   90 |    NULL |  
| 100009 | Linda     | King     | ARTS  | CSCI  |  125 |    1018 |  
| 100111 | Cathy     | Johanson | NULL  | NULL  |    0 |    1018 |  
+--------+-----------+----------+-------+-------+------+---------+  
11 rows in set (0.001 sec)  
  
MariaDB [toyu]> SELECT \* FROM enroll;  
+--------+---------+-------+----------+  
| stuId  | classId | grade | n\_alerts |  
+--------+---------+-------+----------+  
| 100000 |   10000 | A     |        0 |  
| 100001 |   10000 | NULL  |     NULL |  
| 100002 |   10000 | B-    |        3 |  
| 100000 |   10001 | A     |        2 |  
| 100001 |   10001 | A-    |        0 |  
| 100000 |   10002 | B+    |        1 |  
| 100002 |   10002 | B+    |        2 |  
| 100000 |   10003 | C     |        0 |  
| 100002 |   10003 | D     |        4 |  
| 100004 |   10003 | A     |        0 |  
| 100005 |   10003 | NULL  |     NULL |  
| 100000 |   10004 | A-    |        1 |  
| 100004 |   10004 | B+    |     NULL |  
| 100005 |   10004 | A-    |        0 |  
| 100006 |   10004 | C+    |     NULL |  
| 100005 |   10005 | A-    |        0 |  
| 100006 |   10005 | A     |     NULL |  
| 100005 |   10006 | B+    |     NULL |  
| 100007 |   10007 | F     |        4 |  
| 100008 |   10007 | C-    |        0 |  
| 100007 |   10008 | A-    |        0 |  
| 100000 |   11001 | D     |        4 |  
+--------+---------+-------+----------+  
22 rows in set (0.000 sec)

***Example:***

Consider the following *instance* of the table department:

+----------+------------------------------+------------+----------+  
| deptCode | deptName                     | schoolCode | numStaff |  
+----------+------------------------------+------------+----------+  
| ACCT     | Accounting                   | BUS        |       10 |  
| ARTS     | Arts                         | HSH        |        5 |  
| CINF     | Computer Information Systems | CSE        |        5 |  
| CSCI     | Computer Science             | CSE        |       12 |  
| ENGL     | English                      | HSH        |       12 |  
| ITEC     | Information Technology       | CSE        |        4 |  
| MATH     | Mathematics                  | CSE        |        7 |  
+----------+------------------------------+------------+----------+  
7 rows in set (0.00 sec)

* The name of the table is 'department'.
* There are currently seven rows in the relation*instance*.
* Each row in the table 'department' has the same columns: deptCode, deptName, schoolCode and numStaff. This is the structure of the relation: the relation *schema*.
* The relation schema for department: department(deptCode, deptName, schoolCode, numStaff).
* Each column has a specific data type. Examples:
  1. deptCode: CHAR(4)
  2. deptName: VARCHAR(30)
  3. schoolCode: CHAR(3)
  4. numStaff: TINYINT
* We say the *domain* of the column deptCode is CHAR(4).
* A domain may be understood as the values allowed by the *data type*.
* Thus, the relation schema and column domains form the 'structure' of the database.
* The structures usually do not change much, just like the structure of a building.
* However, the structure can be changed (just like the structure of a building: remodeling).
* On the other hand, the content of a table (the relation *instance*) can be changed from time to time.

MariaDB [toyu]> desc department;  
+------------+-------------+------+-----+---------+-------+  
| Field      | Type        | Null | Key | Default | Extra |  
+------------+-------------+------+-----+---------+-------+  
| deptCode   | char(4)     | NO   | PRI | NULL    |       |  
| deptName   | varchar(30) | NO   | UNI | NULL    |       |  
| schoolCode | char(3)     | YES  | MUL | NULL    |       |  
| numStaff   | tinyint(4)  | YES  |     | NULL    |       |  
+------------+-------------+------+-----+---------+-------+  
4 rows in set (0.005 sec)  
  
Note that 'KEY' and 'INDEX' have the same meaning in MySQL.

* Key: PRI; primary key/index
* Key: UNI; unique key/index
* key: MUL; multiple value index.

***Example:***

After inserting a new row:

INSERT INTO department VALUES ('PHYS', 'Physics','CSE',3);

MariaDB [toyu]> INSERT INTO department VALUES ('PHYS', 'Physics','CSE',3);  
Query OK, 1 row affected (0.012 sec)

The new relation instance of the table department:

MariaDB [toyu]> SELECT \* from department;  
+----------+------------------------------+------------+----------+  
| deptCode | deptName                     | schoolCode | numStaff |  
+----------+------------------------------+------------+----------+  
| ACCT     | Accounting                   | BUS        |       10 |  
| ARTS     | Arts                         | HSH        |        5 |  
| CINF     | Computer Information Systems | CSE        |        5 |  
| CSCI     | Computer Science             | CSE        |       12 |  
| ENGL     | English                      | HSH        |       12 |  
| ITEC     | Information Technology       | CSE        |        4 |  
| MATH     | Mathematics                  | CSE        |        7 |  
| PHYS     | Physics                      | CSE        |        3 |  
+----------+------------------------------+------------+----------+  
8 rows in set (0.000 sec)  
  
After:

DELETE FROM department WHERE deptCode = 'PHYS';

MariaDB [toyu]> DELETE FROM department WHERE deptCode = 'PHYS';  
Query OK, 1 row affected (0.009 sec)

The relation instance reverts back.

MariaDB [toyu]> SELECT \* from department;  
+----------+------------------------------+------------+----------+  
| deptCode | deptName                     | schoolCode | numStaff |  
+----------+------------------------------+------------+----------+  
| ACCT     | Accounting                   | BUS        |       10 |  
| ARTS     | Arts                         | HSH        |        5 |  
| CINF     | Computer Information Systems | CSE        |        5 |  
| CSCI     | Computer Science             | CSE        |       12 |  
| ENGL     | English                      | HSH        |       12 |  
| ITEC     | Information Technology       | CSE        |        4 |  
| MATH     | Mathematics                  | CSE        |        7 |  
+----------+------------------------------+------------+----------+  
7 rows in set (0.000 sec)

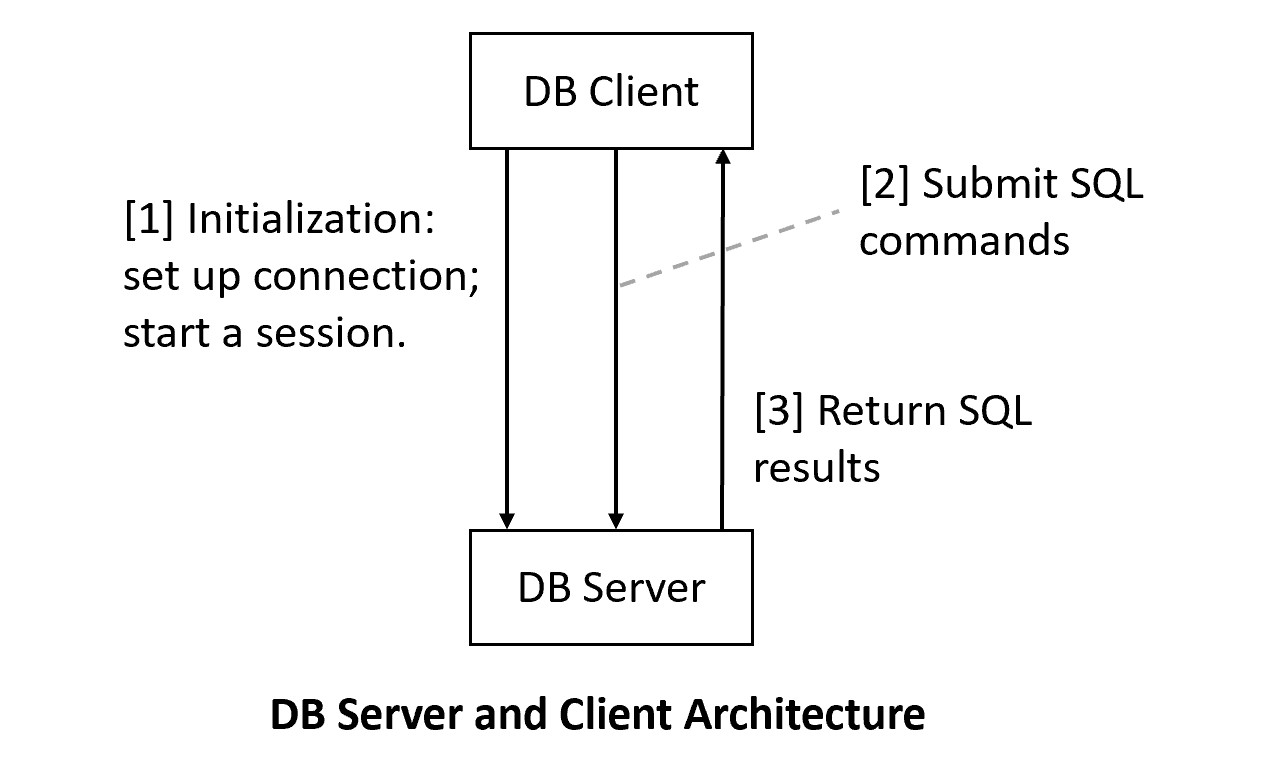
* Note the terms table and relation can be ambiguous. They may mean:
  1. the relation schema: the 'structure' of the table.
  2. the relation instance: actual data in the relation at a specific time.
* Note that relations are a logical concept to reason with.
* Relations must eventually be stored in the file systems to ensure persistence.
* For example, a row *may* be stored as a record, in which a column is a *field* of the record. A row is also known as a *tuple* in the relational model.
* Conceptually, a row may be used to represent (or store information about) an entity, a relationship, or a concept.
  1. A student record may represent a student (entity)
  2. An enrollment record may represent a relationship: a student taking a class.
* Tables may be 'connected' through unique identifiers, known as*foreign keys.* Example:
  1. The enroll table is related to the student table through stuId:

MariaDB [toyu]> SELECT \* FROM student;  
+--------+-----------+----------+-------+-------+------+---------+  
| *stuId*  | fname     | lname    | major | minor | ach  | advisor |  
+--------+-----------+----------+-------+-------+------+---------+  
| 100000 | Tony      | Hawk     | CSCI  | CINF  |   40 |    1011 |  
| 100001 | Mary      | Hawk     | CSCI  | CINF  |   35 |    1011 |  
| 100002 | David     | Hawk     | CSCI  | ITEC  |   66 |    1012 |  
| 100003 | Catherine | Lim      | ITEC  | CINF  |   20 |    NULL |  
| 100004 | Larry     | Johnson  | ITEC  | NULL  |   66 |    1017 |  
| 100005 | Linda     | Johnson  | CINF  | ENGL  |   13 |    1015 |  
| 100006 | Lillian   | Johnson  | CINF  | ITEC  |   18 |    1016 |  
| 100007 | Ben       | Zico     | NULL  | NULL  |   16 |    NULL |  
| 100008 | Bill      | Ching    | ARTS  | NULL  |   90 |    NULL |  
| 100009 | Linda     | King     | ARTS  | CSCI  |  125 |    1018 |  
| 100111 | Cathy     | Johanson | NULL  | NULL  |    0 |    1018 |  
+--------+-----------+----------+-------+-------+------+---------+  
11 rows in set (0.001 sec)  
  
MariaDB [toyu]> SELECT \* FROM enroll;  
+--------+---------+-------+----------+  
| *stuId*  | classId | grade | n\_alerts |  
+--------+---------+-------+----------+  
| 100000 |   10000 | A     |        0 |  
| 100001 |   10000 | NULL  |     NULL |  
| 100002 |   10000 | B-    |        3 |  
| 100000 |   10001 | A     |        2 |  
| 100001 |   10001 | A-    |        0 |  
| 100000 |   10002 | B+    |        1 |  
| 100002 |   10002 | B+    |        2 |  
| 100000 |   10003 | C     |        0 |  
| 100002 |   10003 | D     |        4 |  
| 100004 |   10003 | A     |        0 |  
| 100005 |   10003 | NULL  |     NULL |  
| 100000 |   10004 | A-    |        1 |  
| 100004 |   10004 | B+    |     NULL |  
| 100005 |   10004 | A-    |        0 |  
| 100006 |   10004 | C+    |     NULL |  
| 100005 |   10005 | A-    |        0 |  
| 100006 |   10005 | A     |     NULL |  
| 100005 |   10006 | B+    |     NULL |  
| 100007 |   10007 | F     |        4 |  
| 100008 |   10007 | C-    |        0 |  
| 100007 |   10008 | A-    |        0 |  
| 100000 |   11001 | D     |        4 |  
+--------+---------+-------+----------+  
22 rows in set (0.000 sec)

**3. MySQL**

* The standard query language for RDBMS is Structured Query Language (SQL).
* We use MySQL (or MariaDB) in this class.

DBMS mostly uses a client-server architecture.



**3.1 MySQL Server Setup**

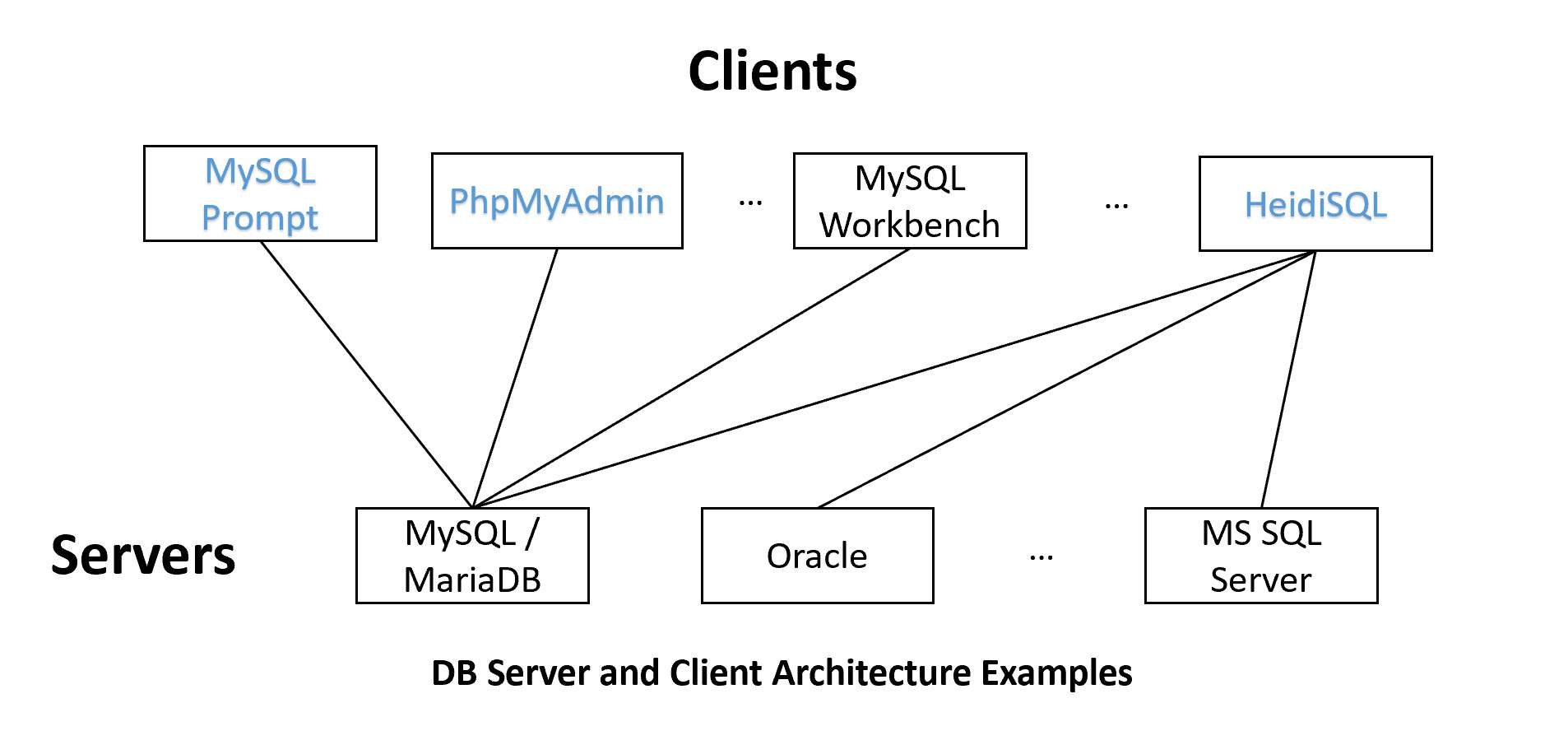
We will use MariaDB that is a part of XAMPP. Do not recommend installing standalone MySQL.

[1] Install XAMPP, which contains many server software configured to work together for development purposes. For XAMPP, we will use MySQL/Maria DB and Apache (Web server).

1. Recommended to install XAMPP in the*top* level: c:\xampp (likely the default).
2. Set up development accounts immediately using *phpMyAdmin* after installation.
3. Change the root password (optional but recommended): a secure step that requires tinkering.
4. To ensure that PHPMyAdmin will work on a new admin account (optional):
   1. Use PHPMyAdmin to create a new admin account, e.g., "frog\_ad", with the password "a\_new\_prince" for both hostname '%' and 'localhost'
   2. PhpMyAdmin uses the default root account (with no initial password) via localhost.
   3. Thus, you will need to supply the new username and password to start up PhpMyAdmin by editing the file c:\xampp\phpMyAdmin\config.inc.php, search change the line to, for example:
      1. $cfg['Servers'][$i]['user'] = 'frog\_ad';
      2. $cfg['Servers'][$i]['password'] = 'a\_new\_prince';

**3.2 MySQL Clients Setup:**

It is common to use multiple clients to connect to a backend database server. In this course, we will use three clients in our classes. You may use your own favorite clients (e.g., MySQL Workbench). However, I may not be as helpful in these clients.



[1] MySQL Command-Line Prompt: will be used in this class.

1. Come with (1) XAMPP/MariaDB or (2) MySQL 8.x. (Note that the two versions of mysql prompt are somewhat different.)
   1. MariaDB mysql: <https://mariadb.com/kb/en/mysql-command-line-client/>
   2. MySQL 8.x mysql: <https://dev.mysql.com/doc/refman/8.0/en/mysql.html>
2. A command line text-based MySQL-specific client.
3. You may set the PATH variable so you can call mysql prompt anywhere, such as by adding "c:\xampp\mysql\bin" in the PATH system environment variable.

***Example:***

**mysql –h *host* -u *user* -p**

or

**mysql –h *host* -u *user* -p -P port\_number**

[2] PhPMyAdmin

1. A Web-based GUI client focused on DB administration.
2. After starting both MySQL and Apache in XAMPP, go to localhost in your browser.
3. MySQL specific.

[3] HeidiSQL: will be used in this class

1. A general Windows GUI SQL client

[4] MySQL Workbench:

1. A GUI MySQL client that comes with MySQL 8.x (but not XAMPP)

**4. SQL**

* Note that different DBMS support different versions of SQL. They may not fully support the standards and may include extensions.
* For example, older versions of MySQL do not support EXCEPT, which is in ANSI SQL 92.
* When developing databases in a given DBMS, portability and backward compatibility issues are a consideration for your choice of SQL statements.
* MySQL is*not* based on a pure relational model. For example:
  1. Query results may contain duplicate rows.
  2. Order may be important.
* SQL is easy to start with. Not so easy to become an expert.
* SQL is a rich and essentially declarative but with some procedural constructs.
* For the time being, we only need to know the basic form of the SELECT statement to get some tastes of SQL and RDBMS.

SELECT DISTINCT <<result\_columns>>  
FROM <<source\_tables>> -- conceptually joined to form a large table  
WHERE <<conditions\_for\_inclusion>>

**Declarative Analysis**

1. <<source\_tables>>: the source tables to gather the result data
2. <<conditions\_for\_inclusion>>: the conditions to be satisfied for results to be included and the conditions the tables should be connected together.
3. <<result\_columns>>: the result columns or expressions desired to be displayed.

***Examples:***

Using toyu, executing the SQL:

-- Department codes and their names  
SELECT DISTINCT deptCode, deptName  
FROM department;  
  
-- Faculty information from the department 'CSCI'  
SELECT DISTINCT \*  
FROM faculty  
WHERE deptCode = 'CSCI';  
  
-- Faculty names from the department 'CSCI'  
SELECT DISTINCT fname, lname  
FROM faculty  
WHERE deptCode = 'CSCI';  
  
-- Associate professor names from the department 'CSCI'.  
SELECT DISTINCT fname, lname  
FROM faculty  
WHERE deptCode = 'CSCI'  
AND `rank` = 'Associate Professor';  
  
-- Department names and numbers of faculty with the numbers  
SELECT DISTINCT deptName, numStaff  
FROM department  
WHERE numStaff >= 10;  
  
-- Names of all faculty members and their  
-- department names and ranks.  
SELECT DISTINCT faculty.fName, faculty.lname,  
   department.deptName, faculty.`rank`  
FROM department, faculty  
WHERE faculty.deptCode = department.deptCode;  
  
SELECT DISTINCT faculty.fName, faculty.lname,  
   department.deptName, faculty.`rank`  
FROM department INNER JOIN faculty ON (faculty.deptCode = department.deptCode);

Result:  
  
MariaDB [toyu]> -- Department codes and their names  
MariaDB [toyu]> SELECT DISTINCT deptCode, deptName  
    -> FROM department;  
+----------+------------------------------+  
| deptCode | deptName                     |  
+----------+------------------------------+  
| ACCT     | Accounting                   |  
| ARTS     | Arts                         |  
| CINF     | Computer Information Systems |  
| CSCI     | Computer Science             |  
| ENGL     | English                      |  
| ITEC     | Information Technology       |  
| MATH     | Mathematics                  |  
+----------+------------------------------+  
7 rows in set (0.000 sec)  
  
MariaDB [toyu]>  
MariaDB [toyu]> -- Faculty information from the department 'CSCI'  
MariaDB [toyu]> SELECT DISTINCT \*  
    -> FROM faculty  
    -> WHERE deptCode = 'CSCI';  
+-------+--------+--------+----------+---------------------+  
| facId | fname  | lname  | deptCode | rank                |  
+-------+--------+--------+----------+---------------------+  
|  1011 | Paul   | Smith  | CSCI     | Professor           |  
|  1012 | Mary   | Tran   | CSCI     | Associate Professor |  
|  1013 | David  | Love   | CSCI     | NULL                |  
|  1014 | Sharon | Mannes | CSCI     | Assistant Professor |  
+-------+--------+--------+----------+---------------------+  
4 rows in set (0.000 sec)  
  
MariaDB [toyu]>  
MariaDB [toyu]> -- Faculty names from the department 'CSCI'  
MariaDB [toyu]> SELECT DISTINCT fname, lname  
    -> FROM faculty  
    -> WHERE deptCode = 'CSCI';  
+--------+--------+  
| fname  | lname  |  
+--------+--------+  
| Paul   | Smith  |  
| Mary   | Tran   |  
| David  | Love   |  
| Sharon | Mannes |  
+--------+--------+  
4 rows in set (0.000 sec)  
  
MariaDB [toyu]>  
MariaDB [toyu]> -- Associate professor names from the department 'CSCI'  
MariaDB [toyu]> SELECT DISTINCT fname, lname  
    -> FROM faculty  
    -> WHERE deptCode = 'CSCI'  
    -> AND `rank` = 'Associate Professor';  
+-------+-------+  
| fname | lname |  
+-------+-------+  
| Mary  | Tran  |  
+-------+-------+  
1 row in set (0.000 sec)  
  
MariaDB [toyu]>  
MariaDB [toyu]> -- Department names and numbers of faculty with the numbers  
MariaDB [toyu]> SELECT DISTINCT deptName, numStaff  
    -> FROM department  
    -> WHERE numStaff >= 10;  
+------------------+----------+  
| deptName         | numStaff |  
+------------------+----------+  
| Accounting       |       10 |  
| Computer Science |       12 |  
| English          |       12 |  
+------------------+----------+  
3 rows in set (0.002 sec)  
  
MariaDB [toyu]>  
MariaDB [toyu]> -- Names of all faculty members and their  
MariaDB [toyu]> -- department names and ranks.  
MariaDB [toyu]> SELECT DISTINCT faculty.fName, faculty.lname,  
    -> department.deptName, faculty.`rank`  
    -> FROM department, faculty  
    -> WHERE faculty.deptCode = department.deptCode;  
+----------+----------+------------------------------+---------------------+  
| fName    | lname    | deptName                     | rank                |  
+----------+----------+------------------------------+---------------------+  
| Paul     | Smith    | Computer Science             | Professor           |  
| Mary     | Tran     | Computer Science             | Associate Professor |  
| David    | Love     | Computer Science             | NULL                |  
| Sharon   | Mannes   | Computer Science             | Assistant Professor |  
| Daniel   | Kim      | Computer Information Systems | Professor           |  
| Andrew   | Byre     | Computer Information Systems | Associate Professor |  
| Deborah  | Gump     | Information Technology       | Professor           |  
| Art      | Allister | Arts                         | Assistant Professor |  
| Benjamin | Yu       | Information Technology       | Lecturer            |  
| Katrina  | Bajaj    | English                      | Lecturer            |  
| Jorginlo | Neymar   | Accounting                   | Assistant Professor |  
+----------+----------+------------------------------+---------------------+  
11 rows in set (0.000 sec)  
  
MariaDB [toyu]>  
MariaDB [toyu]> SELECT DISTINCT faculty.fName, faculty.lname,  
    -> department.deptName, faculty.`rank`  
    -> FROM department INNER JOIN faculty ON (faculty.deptCode = department.deptCode);  
+----------+----------+------------------------------+---------------------+  
| fName    | lname    | deptName                     | rank                |  
+----------+----------+------------------------------+---------------------+  
| Paul     | Smith    | Computer Science             | Professor           |  
| Mary     | Tran     | Computer Science             | Associate Professor |  
| David    | Love     | Computer Science             | NULL                |  
| Sharon   | Mannes   | Computer Science             | Assistant Professor |  
| Daniel   | Kim      | Computer Information Systems | Professor           |  
| Andrew   | Byre     | Computer Information Systems | Associate Professor |  
| Deborah  | Gump     | Information Technology       | Professor           |  
| Art      | Allister | Arts                         | Assistant Professor |  
| Benjamin | Yu       | Information Technology       | Lecturer            |  
| Katrina  | Bajaj    | English                      | Lecturer            |  
| Jorginlo | Neymar   | Accounting                   | Assistant Professor |  
+----------+----------+------------------------------+---------------------+  
11 rows in set (0.000 sec)

***Classroom Demonstration and Exercises:***

***Example***. Show all student names.

Expected Result:

+---------+-----------+  
| lname   | fname     |  
+---------+-----------+  
| Hawk    | Tony      |  
| Hawk    | Mary      |  
| Hawk    | David     |  
| Lim     | Catherine |  
| Johnson | Larry     |  
| Johnson | Linda     |  
| Johnson | Lillian   |  
| Zico    | Ben       |  
| Ching   | Bill      |  
| King    | Linda     |  
+---------+-----------+  
10 rows in set (0.00 sec)

Declarative Analysis:

[1] Sources: student  
[2] Conditions: none  
[3] Output fields: lname, fname

SELECT DISTINCT s.fname, s.lname  
FROM student AS s -- s is the alias of student

* Using alias in generally is considered good style.
* Conceptually "FROM student AS s " may be conceptually understood as that s is a row in the table student.

***Example.*** List the last names and first names of students minoring in CINF and having 1011 as faculty advisor.

+-------+-------+  
| lname | fname |  
+-------+-------+  
| Hawk  | Tony  |  
| Hawk  | Mary  |  
+-------+-------+  
2 rows in set (0.001 sec)

[Analysis]

[1] Source tables: student

[2] Conditions:

1. minor = 'CINF'
2. advisor = 1011

[3] Output columns:

1. lname
2. fname

[suggested solution of sample question]  
  
SELECT DISTINCT s.fname, s.lname  
FROM student AS s -- s is the alias of student  
WHERE s.minor = 'CINF'  
AND s.advisor = 1011;

***Classroom examples:***

1. All student names and the major codes.  
2. All student names and the major department names.  
3. All student names enrolled in the class with id 10003.  
4. Show all information of students majoring in ‘MATH’.  
5. Show the names and credits of students majoring in 'CSCI'.  
6. Show the names and credits of students majoring in 'CSCI' and having 40 or more ach credits.  
7. Show the id of students enrolled in the course CSCI 4333.  
8. Show the code of departments with faculty in the rank of 'Professor'.  
9. Show the names of departments with faculty in the rank of 'Professor'.  
10. Show the names of students who have enrolled in the course CSCI 4333.  
11. Show the names and major names of every student.  
12. Show the names, major names, and advisor names of every student.  
13. Show the student names and their major names for all students who have received a grade A in a class offered by a faculty from the CSCI department.  
14. Show the student names who have enrolled in at least two classes.

Will work on past SQL assignments.