**CSC 365-07: Introduction to Database Systems** **Spring 2023**

# Lecture 1: Database and DBMS

*Instructor: Alex Dekhtyar* *Ishaan Sathaye*

## Introduction

Definition of a database and DBMS in Professor Notes.

**CSC 365-07: Introduction to Database Systems** **Spring 2023**

# Lecture 2: Relational Data Model

*Instructor: Alex Dekhtyar* *Ishaan Sathaye*

## Relational Data Model

**Definition 1** *Relational data model is an approach to organizing collections of data*

- Relation
  - Relational Table $\longrightarrow$ **Name + Schema**
    - Schema: List of attribute name + attribute type pairs
- Relational Database $\longrightarrow$ **Collection of Relations tables**
- **Table Instance**: set of records with instantiated values of the attributes
  - Finite
  - Records, rows, tuples

One unit of data is called a **datum**.

Object, entity, event: description of one object, entity, event

- **Records** consist of attributes or fields (rows in the table).
- **Attributes** is a named container for a value of a specific type.

### Database Table Constraint

**Definition 2** *Limitations of table instances*

- **Candidate Key**: set or lists of attributes that uniquely define a record in a table, **minimal such set of attributes**, made up of multiple attributes sometimes.
  - **Every attribute is necessary.**

## Examples

### CSC 365 Example

Course Object:

- Prefix: CSC $\longrightarrow$ **String**
- Course #: 365 $\longrightarrow$ **Integer**
- Name: Introduction to Database Systems $\longrightarrow$ **String**
- Description: Basic Principles, ... $\longrightarrow$ **String**
- Units: 4 $\longrightarrow$ **Integer**

Department Object:

- Name: Computer Science and Software Engineering
- Abbreviation: CSSE
- Building: 14
- Room: 245
- College: CENG

Stringing these objects together based on relationship would make a **network model**.

## Schema Example

```
Course(Prefix String, Course# Integer, Name String, Description
String, Units Integer)
```

| Prefix | Course# | Name | Description | Units |
|---|---|---|---|---|
| CSC | 365 | Introduction to Database Systems | Basic Principles, ... | 4 |
| CSC | 357 | Systems Programming | ... | 4 |

```
Department(Name, College, Building, Room): Department would also have a table as well.
```

**CSC 365-07: Introduction to Database Systems** **Spring 2023**

# Lecture 3: RDM Cont.

*Instructor: Alex Dekhtyar* *Ishaan Sathaye*

## Relational Data Model

What makes a record unique?

- **Superkey**: any set of attributes that uniquely defines a record in a table
- **Primary Key**: candidate key chosen by you

**CSC 365-07: Introduction to Database Systems** **Spring 2023**

# Lecture 4: SQL DDL and DML

*Instructor: Alex Dekhtyar* *Ishaan Sathaye*

## MySQL Access

1. Server Address = host: **mysql.labthreesixfive.com**
2. Port: 3306
3. username
4. password

MySQL Database

- Namespace
- Collection of Tables
- Set of Permissions

## Case Sensitivity

### Case Sensitive

- Table Names
- Database Names

### Not Case Sensitive

- Attribute Names
- SQL Keywords

## Types

- **Numeric Types**
  - **Integer Types**
    - TINYINT
    - SMALLINT
    - MEDIUMINT

    - **INT**
    - BIGINT
  - **Floating Point Types**
    - **FLOAT**
    - **DOUBLE(P, D)**
    - **DECIMAL**

- **String Types**
  - **Character Types**
    - **CHAR(N)** $\longrightarrow$ **Fixed Length**
    - **VARCHAR(N)** $\longrightarrow$ **Variable Length**
    - TINYTEXT
    - **TEXT** $\longrightarrow$ for storing large amounts of text
    - MEDIUMTEXT
    - LONGTEXT

- **Date and Time Types**
  - **Date Types**
    - DATE
    - DATETIME
    - TIMESTAMP
    - TIME
    - YEAR

## Data Definition Language (DDL)

**Commands from DDL act upon the schema**

- CREATE TABLE
- DROP TABLE
- ALTER TABLE

### Define a Relational Table

Aspects needed to define a table:

- Table Name
- Attributes: Name + Type
- Constraints

```
CREATE TABLE <table_name> (
    <attribute_name> <sql_type> [<single_line_constraints>],
    ...,
    <attribute_name> <sql_type> [<single_line_constraints>] [,
    <constraints>[,
    <constraints>]
]);
```

## Data Manipulation Language (DML)

**Commands from DML act upon the instance.**

- INSERT
- DELETE
- UPDATE

### Inserting Data

```
INSERT INTO <table_name>(<attribute_name>, ...)
    VALUES (<value>, ...);
```

Supply values in order of attribute declarations in CREATE TABLE statement. Can omit the attribute names if values supplied are in the same order. If need to omit a value then omit that attribute name as well.

## More on Constraints

- **[NOT] NULL** - attribute cannot be null
- **UNIQUE**
- **PRIMARY KEY**
- **FOREIGN KEY**
- **DEFAULT <exp>** - default value for attribute
- **AUTO_INCREMENT** - means that the attribute is an integer and is automatically incremented

## Lab 2

MySQL Server

- LabThreeSixFive.com

- mysql command line client
- IDE (DatGrip)
- mysql connectivity from Python

Lab 2 uses Create Table, Drop Table, and Insert.

## Code from Lab

```
show tables

CREATE TABLE Departments (

    DeptId INT PRIMARY KEY,
    Abbr VARCHAR(20) UNIQUE, -- UNIQUE makes candidate key
    Name VARCHAR(128) UNIQUE,
    College CHAR(10),
    Building INT,
    Room CHAR(6),
    -- set multiple candidate keys at the bottom
    UNIQUE(Building, Room),
    -- foreign key always a separate line statement:
    -- FOREIGN KEY(College) REFERENCES colleges(abbr)

);

describe colleges;
SELECT * FROM colleges;

show CREATE TABLE colleges;

show CREATE TABLE Departments;

INSERT INTO Departments
    VALUES(1, 'CSSE', 'Computer Science and Software Engineering', 'CENG', 14, '245');

INSERT INTO Departments(DeptId, Abbr, Name, College, Building, Room)
    VALUES(1, 'CSSE', 'Computer Science and Software Engineering', 'CENG', 14, '245');
```

**CSC 365-07: Introduction to Database Systems** **Spring 2023**

# Lecture 5: DDL and DML Continued

*Instructor: Alex Dekhtyar* *Ishaan Sathaye*

## DML

### Updating Data

```
UPDATE <table_name>
    SET <attribute_name> = <value>
    WHERE <condition>;
```

#### Example

```
UPDATE colleges
    SET abbr = 'COSAM'
    WHERE abbr = 'COASM'
```

WHERE clause is a filter that determines which rows are updated.

### Deleting Data

```
DELETE FROM <table_name> -- just this is a valid command to delete all rows
    WHERE <condition>;
```

## DDL

### Altering Tables

```
ALTER TABLE <table_name>
    <Command> <parameters>;
```

#### Commands

- ADD - add a column/attribute/key
- DROP
- MODIFY
- RENAME

**Parameters**

- COLUMN
- CONSTRAINT
- FOREIGN KEY
- PRIMARY KEY
- UNIQUE

Adding an attribute, dropping/adding a constraint, renaming a table, disable/enabling keys, and modifying attributes examples are in this professor notes: `4-SQLDDLDML.pdf`

**CSC 365-07: Introduction to Database Systems** **Spring 2023**

# Lecture 6: DML/DDL Cont., WHERE Clause., and MySQL Conn.

*Instructor: Alex Dekhtyar* *Ishaan Sathaye*

## Announcements

### Running Scripts for Lab 2

Can run from command line using mysql command or using mysql client. For running using mysql command need to specify the database if not using default database.

```
source script.sql
```

## DML/DDL

**Data Manipulation works on instance and Data Definition works on schema.**

### Altering a Table

Modifying the schema. ALTER examples in class.

For CREATE TABLE, you can name constraints:

```
CREATE TABLE Example (
    Id int PRIMARY KEY,
    X INT,
    Y INT,
    CONSTRAINT Point UNIQUE (X, Y)
);
```

### Updating and Deleting from Table: WHERE Clause

Ex. Deleting in Table

```
DELETE FROM test02
    WHERE b > c
```

This deletes rows where b is greater than c.

Ex. Deleting with Scope

```
DELETE FROM test02
```

```
FOR EACH ROW in test01
DO
    DELETE FROM test02 -- delete(row, condition)
    WHERE b > c
```

### SQL Boolean Expressions

- 0, 1
- Builtin: IN(...) $\longrightarrow$ returns bool
- $< Expression > < op1 > < op2 >$ // can also use IN or LIKE
- $< Expression >$ AND $< Expression >$
- $< Expression >$ OR $< Expression >$
- NOT $< Expression >$

## MySQL Connectivity

Breifly went over the Python examples on Course webpage that connect to MySQL server.

**CSC 365-07: Introduction to Database Systems** **Spring 2023**

Lecture 7: Python Connectivity and Relational Algebra

*Instructor: Alex Dekhtyar* *Ishaan Sathaye*

## Python MySQL Connectivity

Relational Database is sitting on a server. It is listening for connections, and our program is a client that connects to the server via the port. Essentially, there is a pipe and a exchange of messages that is happening. Generates a connection object that stores info about how to properly access the database.

### Package

```
import mysql.connector
```

### Connection

5 Things Needed: Host, Port, Username, Password, Database (sometimes not necessarily)

These get passed to mysql.connector.connect() function. This returns a connection object. is_connected() returns a cursor object.

Cursor object that is returned from the connection object. Cursor object is used to execute queries.

## Relational Algebra

Relational $\longrightarrow$ Database Model $\longrightarrow$ Relational Model. Algebra: set of elements & operations on elements

Relational Algebra is operations on relational tables.

Boolean Algebra introduces operations on truth values

- T, F
- ˜, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$

### Notation

Upper case letters like R, S, T, $R_1$, $S_7$, ... are relational table names. Letters from first half of alphabet like A, B, C, ... are attributes names. R($A_1$,..., $A_n$) are to represent schema. t, s, r $\in$ R are tuples. $a_1$, $a_2$, ... are values. Ex. t = ($a_1$,..., $a_n$) and it could be referred to as t.$A_1$ = $a_1$.

## Operations

### Binary

### Unary

- Selection $\sigma$ - filter rows
- Projection $\pi$ - filter columns

## Selection Operation

- $\sigma_{<selectioncondition>}(R)$ - returns rows that satisfy condition
- Selection Condition denoted by $C$
- Ex. C = $A_2 =' Riley' \wedge A_3 =' Hicks'$
- **Formal Notation:** $\sigma_C(R) = \{t \in R | tsatisfiesC\}$

## Projection Operation

- $\pi_{<attributelist>}(R)$ - returns columns that are in attribute list
- $F$ is the projection list which is a list of attributes
- Ex. F = $(B_1, ..., B_m)$ where $B_i \in A_1, ..., A_n$
- **Formal Notation:** $\pi_F(R) = \{t' | \exists, t \in R, s.t. \forall B \in F, t'.B = t.B\}$
- **Projection squeezes out duplicates**