Database Management Systems:
Fundamentals and Introduction to SQL

Intro to MySQL

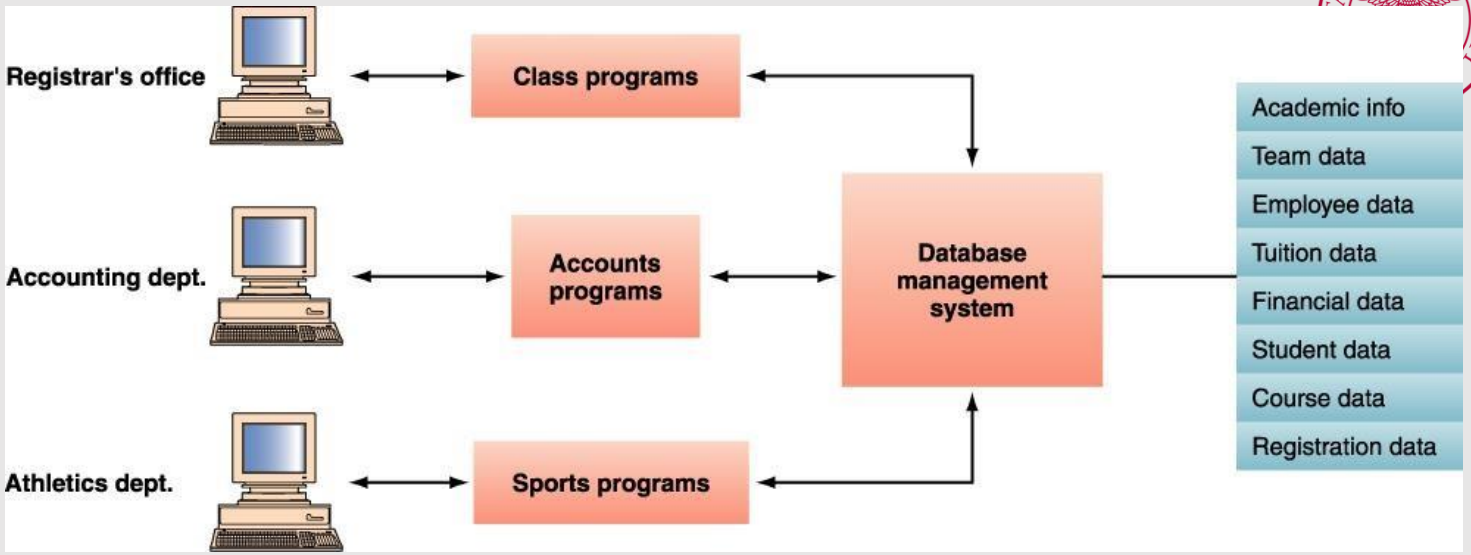
RDBMS

MySQL DBMS

- Client-server architecture
  - Server program: manipulate databases, installed anywhere

  - Client program: communicate between the server and the user, installed locally

Queries in DBMS



- Consider following questions:
  - What is the name of the student with student ID 123456?
  - How many students are enrolled in MSIS 1234?
  - What fraction of students in MSIS 1111 received a grade better than B-?

- RDBMS specialized language – Query language

SQL: Structured Query Language

- SQL is a standard language for storing, manipulating, and retrieving data in **relational databases**.
- RDBMS: A database management system that manages data as a collection of tables in which all relationships are represented by **common values in related tables**

SQL Queries in DBMSs

- In most cases, SQL queries for various RDBMSs are exactly the same.
- In a few cases, there are some syntax differences.

Create a customer table in MySQL:

```
CREATE TABLE customer (
        cust_id int PRIMARY KEY,
        branch varchar(255),
        status varchar(255)
);
```

Create a customer table in Oracle:

```
CREATE TABLE customer (
                cust_id int,
                branch varchar(255),
                status varchar(255),
                CONSTRAINT cust_pk PRIMARY
                        KEY  (cust_id)
);
```

SQL Queries in MySQL: Some Tips

- always use a terminator (;) to denote the end of the statement
- Writing comments
  - Lines beginning with # or -- are ignored as comments. Blank lines are ignored, too.
  - If you want to add comments in several lines, you can use

    /* ……

    ….

    …. */

- Test for a few statements:
  - SELECT NOW();
  - SHOW databases;

Create Database/Schema

- In MySQL, physically, a schema is synonymous with a database
  - CREATE SCHEMA or
  - CREATE DATABASE
- Oracle, DB2, and other enterprise-level database solutions:
  - A Schema is a collection of Tables
  - A Database is a collection of Schemas

SQL Queries Categories

- Data Definition Language (DDL)
  - CREATE
  - DROP
  - ALTER
- Data Manipulation Language (DML)
  - SELECT
  - INSERT
  - DELETE
  - UPDATE
- Data Control Language (DCL) and others
  - Views, indexes, constraints, triggers, transactions, authorizations…

Create a Database

- CREATE DATABASE [IF NOT EXISTS] database_name;
  - [] is for optional statement
- DROP DATABASE database_name;

!!!Be very careful when you Drop a Database

Create a Table Syntax

- USE database_name; # start with designation of database every time

- CREATE TABLE [IF NOT EXISTS] table_name

  (var1 var1_type PRIMARY KEY,

  var2 var2_type,

  ……);

- INSERT ……

Guidelines for Creating Tables

- Identify **data types** for attributes
- Identify attributes that can and **cannot be null**
- Identify **primary key**
- Identify **foreign keys**
- Determine desired **default values** for attributes
- Identify any **domain constraints** for attributes
- Create table and any **indexes**

Create a Table/Tables

```
CREATE TABLE if not exists Customer(
        CustomerID          int                 primary key,
        Customer_Name       varchar(20) not null unique,
        Status              varchar(20) not null
);
CREATE TABLE if not exists Billing(
        InvoiceNo           int                 primary key,
        CustomerID          int         not null,
        Amount              int         not null,
        foreign key CustomerID_fk(customerID)
                references Customer(customerID)
                on update [cascade/restrict/no action]
                on delete [cascade/restrict/no action]
);
```

Customer

CustomerID	Customer Name	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Billing

InvoiceNo	CustomerID	Amount
1	1	$100
2	1	$200
3	2	$150

Create Tables: Defining Attributes and Data Types

```
CREATE TABLE if not exists Customer(
        CustomerID      int             primary key,
        Customer_Name   varchar(20) not null    unique,
        Status          varchar(20) not null
);
CREATE TABLE if not exists Billing(
        InvoiceNo       int             primary key,
        CustomerID      int     not null,
        Amount          int     not null,
        foreign key CustomerID_fk(customerID)
                references Customer(customerID)
                on update [cascade/restrict/no action]
                on delete [cascade/restrict/no action]
);
```

Customer

CustomerID	Customer Name	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Billing

InvoiceNo	CustomerID	Amount
1	1	$100
2	1	$200
3	2	$150

Create Tables: Identifying Foreign Keys and Establishing Relationships

```
CREATE TABLE if not exists Customer(
        CustomerID          int                 primary key,
        Customer_Name       varchar(20) not null    unique,
        Status              varchar(20) not null
);
CREATE TABLE if not exists Billing(
        InvoiceNo           int                 primary key,
        CustomerID          int         not null,
        Amount              int         not null,
        foreign key CustomerID_fk(customerID)
                references Customer(customerID)
                on update [cascade/restrict/no action]
                on delete [cascade/restrict/no action]
);
```

Customer

CustomerID	Customer Name	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Billing

InvoiceNo	CustomerID	Amount
1	1	$100
2	1	$200
3	2	$150

Create Tables: Adding Data Integrity Constraints

```
CREATE TABLE if not exists Customer(
        CustomerID      int                 primary key,
        Customer_Name   varchar(20) not null unique,
        Status          varchar(20) not null
);
CREATE TABLE if not exists Billing(
        InvoiceNo       int                 primary key,
        CustomerID      int         not null,
        Amount          int         not null,
        foreign key CustomerID_fk(customerID)
                references Customer(customerID)
                on update [cascade/restrict/set null]
                on delete [cascade/restrict/set null]
);
```

Customer

CustomerID	Customer Name	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Billing

InvoiceNo	CustomerID	Amount
1	1	$100
2	1	$200
3	2	$150

https://dev.mysql.com/doc/refman/8.0/en/create-table-foreign-keys.html

Insert Values

- Adds one or more rows to a table

Customer

CustomerID	Customer Name	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Billing

InvoiceNo	CustomerID	Amount
1	1	$100
2	1	$200
3	2	$150

```
INSERT INTO Customer(CustomerID, Customer_Name, Status)
VALUES (1, 'Google', 'Active');
……
```

Removing Tables

Command: To drop a table from a database schema.

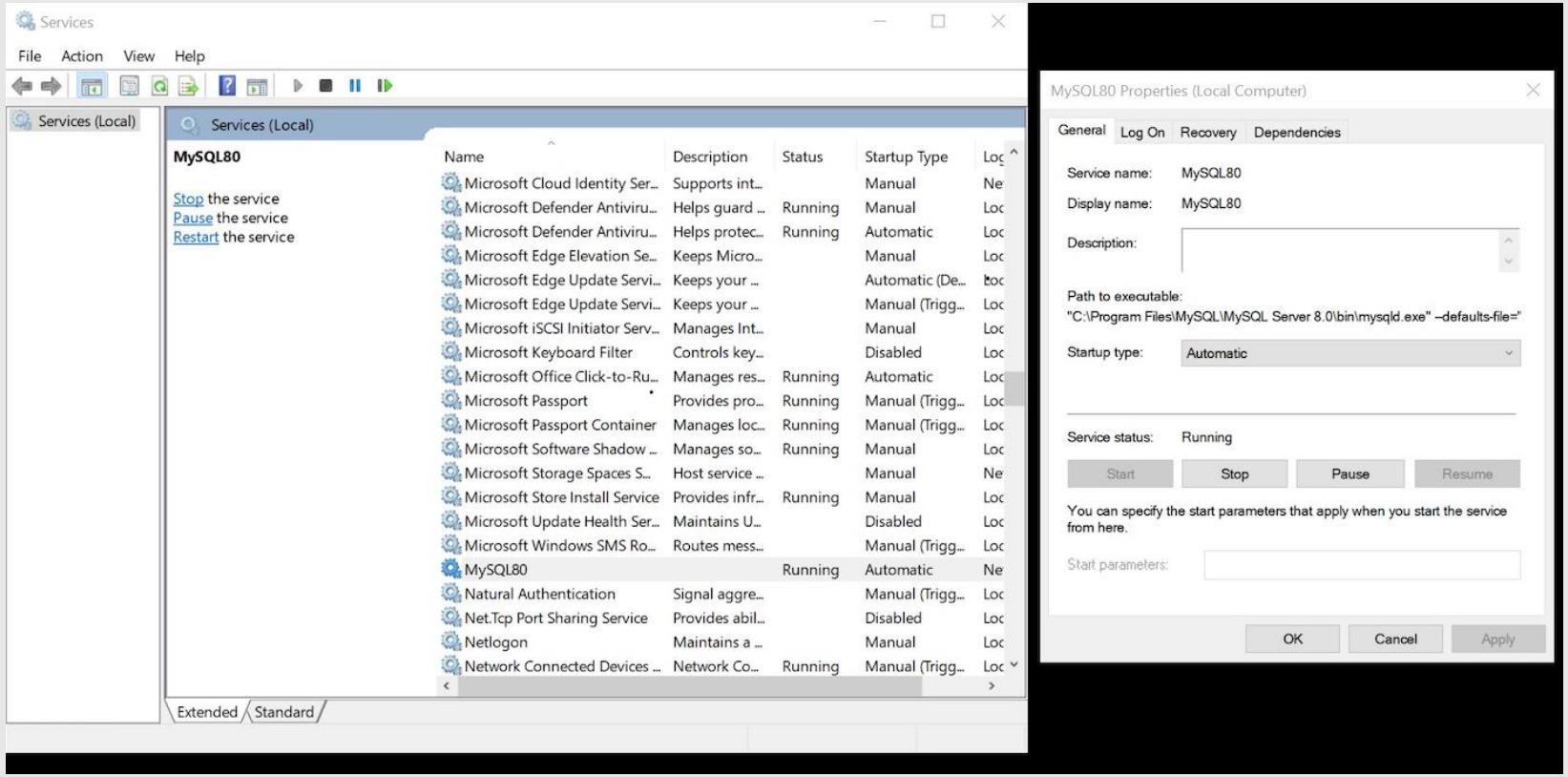
```
DROP TABLE Customer_T;
```

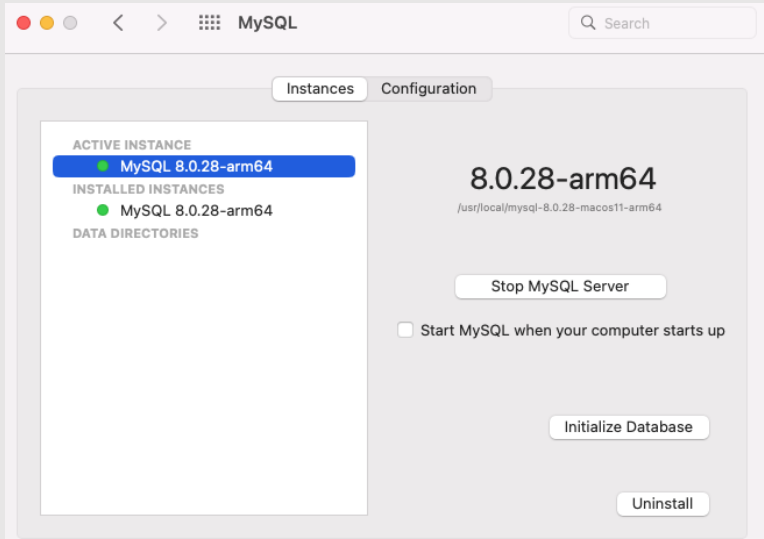
- This command will drop the table and save any pending changes to the database
- To drop a table, you must either own the table or have been granted the DROP ANY TABLE system privilege
- Dropping a table will also cause associated indexes and privileges granted to be dropped
- Many RDBMSs allow users to retain the table's structure but remove all of the data that have been entered in the table with its TRUNCATE TABLE command
- Commands for updating and deleting part of the data in a table: **INSERT, UPDATE, DELETE**

Demo: Work with MySQL server and Client

MySQL Server Status

- Before starting a connection on client, make sure your MySQL server is running





Access to MySQL Client

- MacOS:
  - Terminal
    - `/usr/local/mysql/bin/mysql -h localhost -u root -p`
- Windows:
  - Command line/MySQL 8.0 command line client
    - `"C:\Program Files\SQL\MySQL Server 8.0\bin\mysql" -h localhost -u root -p`
- Via MySQL workbench
  - Connection

Demo: Create Database and Table using MySQL workbench functions and SQL statements

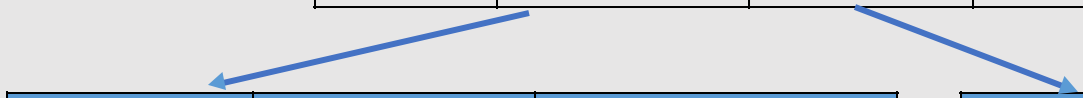
Save Queries

- save as .sql document

Assignment 1

order

order_id	customer_id	item_id	quantity
1	2	1	1
2	2	2	3
3	3	3	5



customer_id	name	email
1	Rosalyn Rivera	rr@adatum.com
2	Jayne Sargen	jayne@test.com
3	Dean Luong	dean@test.com

customer

item_id	name	Price
1	Chair	200
2	Table	100
3	Lamp	50

item