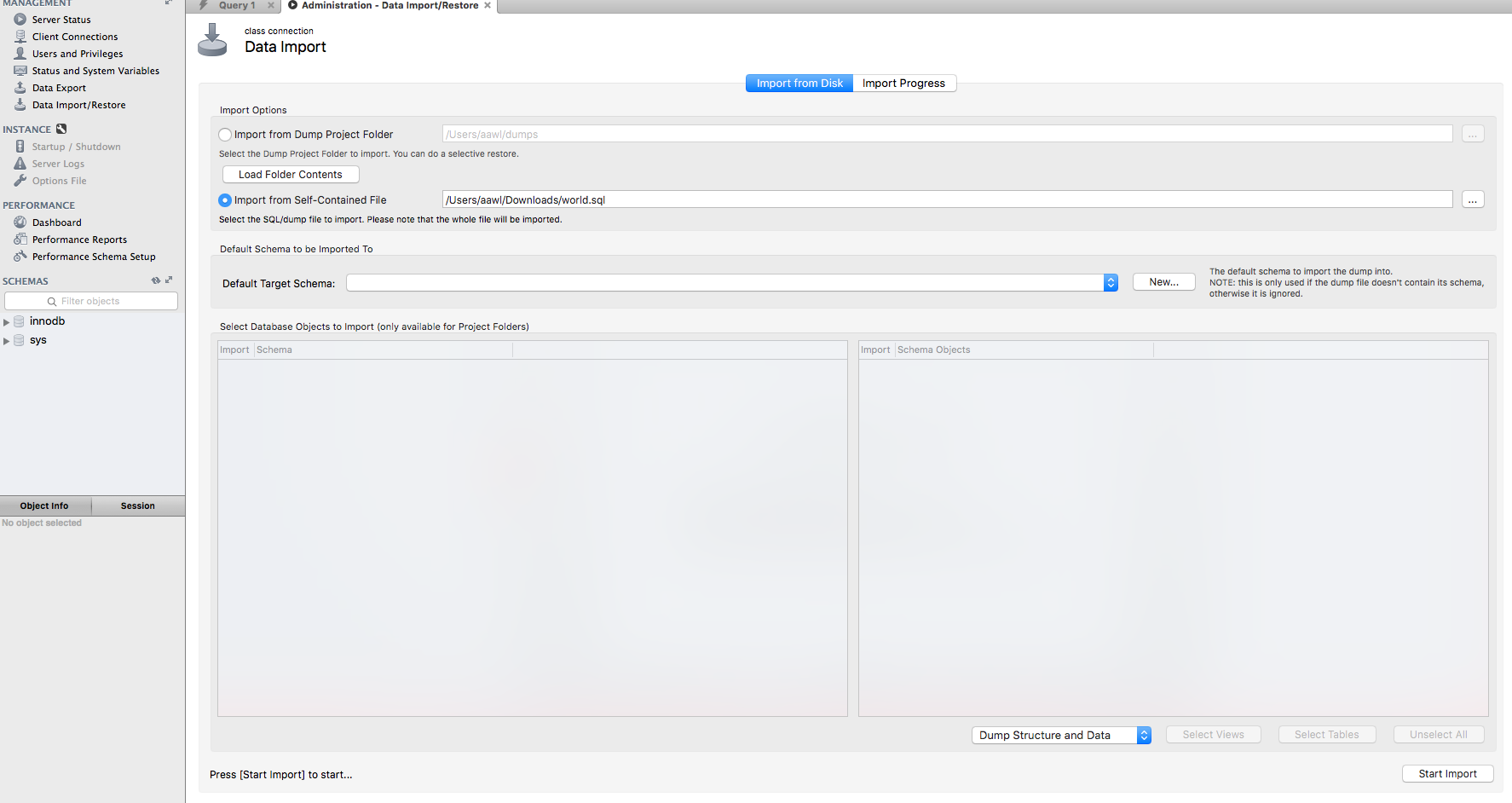
**Lab 1 – MySQL**

**Lab Objectives:**

* + Learn how to import data into MySQL using MySQL Workbench
  + Learn how to create tables in MySQL
    - Primary and foreign keys
    - Autoincrement
    - Create table with a SELECT statement
    - Create table with indexes
  + Get familiar with basic commands in MySQL
  + Learn how to write simple queries in MySQL

**Import Sample World and World-x Database**

1. Log in to MySQL Workbench
2. Click on “Data Import/Restore” icon on top left
3. Download world.sql and world\_x.sql file from Blackboard
4. Select “import from self-contained File” option, browse your computer and select “world.sql” (repeat process for world-x.sql as well) file you just downloaded from Blackboard
5. Click “Start import” icon on bottom right
6. You should see new database under schema after you click on refresh button



1. **Getting Information about a database**

**--select database version and today’s date** (cntr+enter or command+return will execute current statement)

SELECT version(), current\_date;

**-- show time**

SELECT now();

**--show all databases on the server**

Show databases;

**--invoke a specific database (world)**

use world;

**--list tables in this database**

Show tables;

**--invoke world\_x database**

Use world\_x;

**--list tables in this database**

Show tables;

**--describe table**

describe city;

**--show columns from table city**

show columns from city

**--Show create table**

show create table city

1. **Simple outputs**

-- **Output simple Hello World**

SELECT 'Hello, World';

**-- Simple addition**

SELECT 1+3;

**-- Simple multiplication**

SELECT 10\*30;

1. **Working with Rows**

**-- Total number of countries**

SELECT COUNT(\*) FROM country;

**-- All countries**

SELECT \* FROM country;

**-- All countries in alphabetical order**

SELECT \* FROM country ORDER BY Name;

**-- List the first 5 countries in alphabetical order**

SELECT \* FROM country ORDER BY Name limit 5;

-- List 10 countries from row 6 based on alphabetical order (OFFSET 5 will skip first 5 results of the query)

**SELECT \* FROM country ORDER BY Name limit 10 offset 5;**

1. **Working with Columns**

**-- Country name and life expectancy rate ---**

Use world

SELECT Name, LifeExpectancy FROM country;

**-- Country name ISO code, region and population**

SELECT Name AS Country, Code AS ISO, Region, Population AS Pop FROM country;

**-- City name, country code and population**

SELECT Name, CountryCode, Population from city;

**-- Country name, code, region and population/1000**

SELECT Name AS Country, Code AS ISO, Region, Population / 1000 AS 'Population (1000s)' FROM country;

1. **Working with Order By**

**-- Country name and life expectancy in descending order**

SELECT Name, LifeExpectancy AS 'Life Expectancy' FROM country ORDER BY LifeExpectancy DESC;

**-- Country Name, ISO code, region and population ordered by ISO Code**

SELECT Name AS Country, Code AS ISO, Region, Population FROM country ORDER BY Code;

**-- City name, country code and population ordered by city name**

SELECT Name, CountryCode, Population from city ORDER BY Name;

**-- Country name, continent and region and order by continent, region and name**

SELECT Name, Continent, Region FROM country ORDER BY Continent DESC, Region, Name;

1. **Working with WHERE clause**

**-- List of countries with population greater than 100 million**

SELECT Name, Continent, Population FROM country WHERE Population > 100000000 ORDER BY Population DESC;

**-- List of countries that have population less than 100,000**

SELECT Name, Continent, Population FROM country WHERE Population < 100000 ORDER BY Population DESC;

**-- List official languages of China**

SELECT language FROM countrylanguage WHERE Countrycode = 'chn'

1. **Working With Multiple Tables**

**-- List all countries and corresponding cities**

SELECT co.Name as Country, c.Name as City

FROM country co, city c

WHERE co.Code = c.CountryCode

ORDER BY Country, city;

SELECT co.Name as Country, c.Name as City

FROM country co

INNER JOIN city c on co.Code = c.CountryCode

ORDER BY Country, City;

1. **Working With Like and IN ---**

**-- Country names that start with United --**

SELECT Name, Continent, Population FROM country WHERE Name LIKE 'United%' ORDER BY Name;

**-- Country names that end with Island --**

SELECT Name, Continent, Population FROM country WHERE Name LIKE '%island' ORDER BY Name;

**-- All countries that are in Africa and North America continents --**

SELECT Name, Continent FROM country WHERE Continent IN ( 'Africa', 'North America' );

1. **Creating New Tables, Dropping Tables, Creating Tables with Primary and Foreign Keys, Creating Tables using SELECT statement.**

**Use world – make sure to use this database, not world\_x**

**-- Create new table with two columns Name and Age --**

CREATE TABLE People (Name Text, Age INT);

**-- Insert values to the new table --**

INSERT INTO People VALUES ('Michael', 35);

INSERT INTO People VALUES ('John', 15);

INSERT INTO People VALUES ('Ron', 55);

**-- Query the People table --**

SELECT \* FROM People;

--**Create two new tables CUSTOMER and DRINK\_ORDER. Notice how we create Primary and Foreign Keys and how we auto increment ID column.**

CREATE TABLE customer (

id INTEGER auto\_increment,

first\_name VARCHAR(100),

last\_name VARCHAR(100),

address VARCHAR(100),

city VARCHAR(100),

state VARCHAR(2),

zip\_code VARCHAR(10),

PRIMARY KEY (id)

);

CREATE TABLE drink\_order (

id INTEGER auto\_increment,

customer\_id INTEGER,

drink\_description VARCHAR(100),

PRIMARY KEY (id),

CONSTRAINT fk\_drink\_order\_customer

FOREIGN KEY (customer\_id)

REFERENCES customer (id)

);

insert into customer values (null, 'Michael', 'Weston', '123 Brickel', 'Miami', 'FL', '33123');

insert into drink\_order values (123344, 1, 'Scotch');

select \* from customer;

select \* from drink\_order;

--**Create table using existing table with SELECT** (select count of all rows from *an existing* table COUNTRY; create new table COUNTRY2 using a SELECT statement, getting only country names which start with letter A; select from this new table COUNTRY2)

use world

select count(\*) from country

CREATE TABLE country2 as (select \* from country where name like 'A%');

select \* from country2

--**Dropping an existing table**

Use world

Select \* from country2

DROP TABLE IF EXISTS country2

--you will get an error message here because we no longer have this table

select \* from country2

**Create table with indexes (**Index will allow for faster retrieval of data)

Use world\_x

CREATE TABLE customer (

id INTEGER auto\_increment,

first\_name VARCHAR(100),

last\_name VARCHAR(100),

address VARCHAR(100),

city VARCHAR(100),

state VARCHAR(2),

zip\_code VARCHAR(10),

PRIMARY KEY (id),

INDEX last\_name\_idx (last\_name ASC)

);

CREATE TABLE drink\_order (

id INTEGER auto\_increment,

customer\_id INTEGER,

drink\_description VARCHAR(100),

PRIMARY KEY (id),

CONSTRAINT fk\_drink\_order\_customer

FOREIGN KEY (customer\_id)

REFERENCES customer (id)

);

insert into customer values (null, 'Michael', 'Weston', '123 Brickel', 'Miami', 'FL', '33123');

insert into drink\_order values (123344, 1, 'Scotch');

**Attempt to drop table with foreign keys dependencies**

DROP TABLE IS EXISTS customer

Should get an error code: Cannot delete or update a parent row: a foreign key constraint fails

Solution: first drop the table with constraint pointing to CUSTOMER table and then drop CUSTOMER TABLE

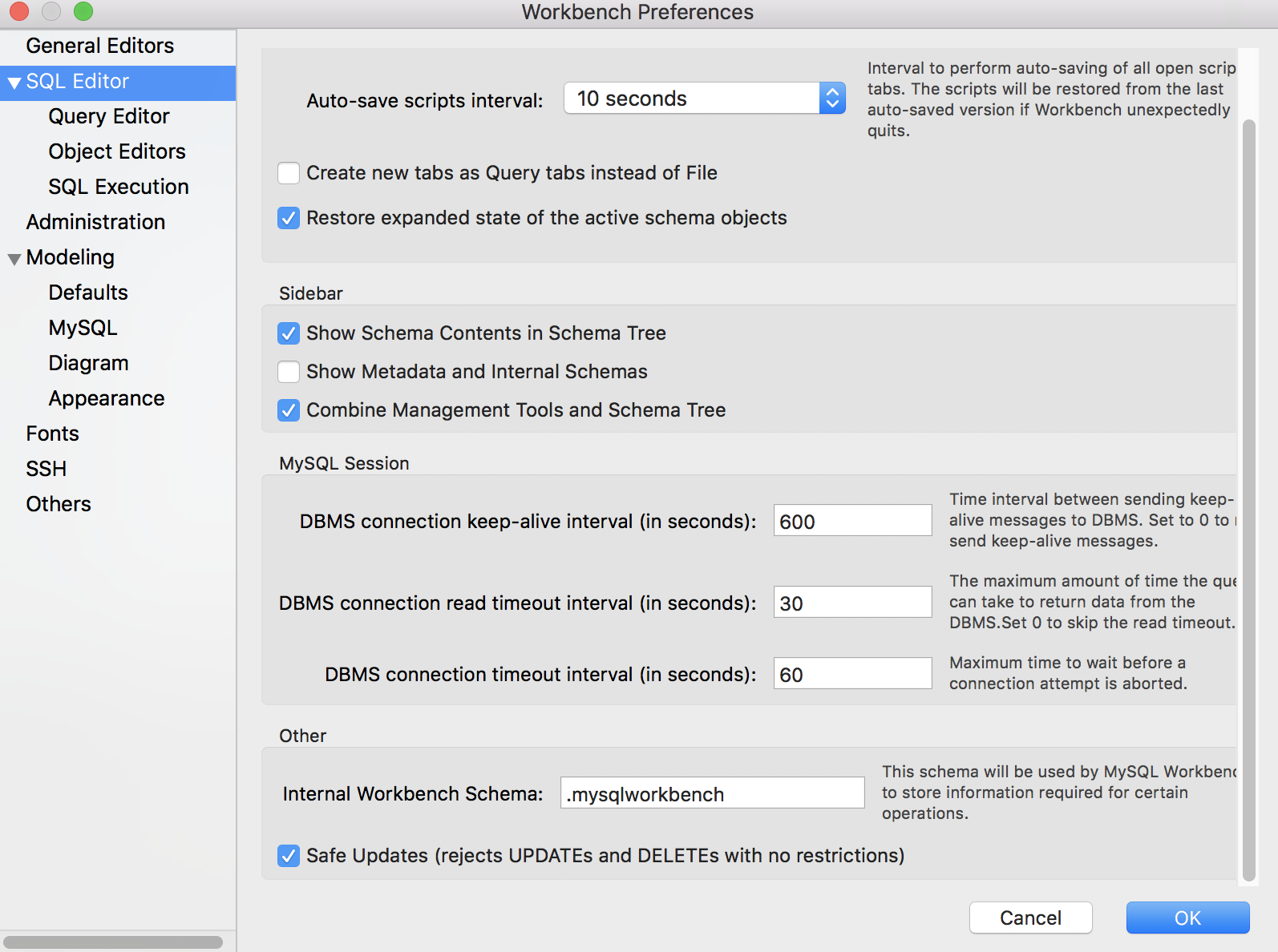
DROP TABLE IF EXISTS drink\_order

DROP TABLE IF EXISTS customer

**10. Updating**

**-- change Workbench Preference /SQL editor and then uncheck safe updates. Reconnect to Workbench**

**MySQL Workbench 🡪 Preferences 🡪 Unckeck Safe Updates 🡪 reconnect**

****

use world

**-- change Michael's age from 35 to 30 --**

UPDATE People SET Age = 30 WHERE Name = 'Michael';

**-- Query the People table after the update --**

SELECT \* FROM People;

**-- change Ron's name to Ronald and his age to 60--**

UPDATE People SET Name = 'Ronald', Age = 60 WHERE Name = 'Ron';

**-- Query the People table after the update --**

SELECT \* FROM People;

1. **Deleting**

**-- Delete John's record --**

DELETE FROM People WHERE Name = 'John';

**-- Query the People table after the delete --**

SELECT \* FROM People;

1. **Using aliases in SELECT**

use world

SELECT code, name FROM country

SELECT code as country\_code, name as country\_name FROM country;

SELECT code as 'Country Code', name as 'Country Name' FROM country;

1. **More Selects with Operators**

use world;

SELECT \* FROM country

--**select countries located in Asia**

SELECT \* FROM country WHERE continent = 'Asia';

--**select countries located in the Middle East**

SELECT \* FROM country WHERE region = 'Middle East';

**--select countries not located in the Middle East**

SELECT \* FROM country WHERE region <> 'Middle East';

**--select countries not located in Asia**

SELECT \* FROM country WHERE continent != 'Asia';

**--select count of countries located in Asia**

SELECT count(\*) FROM country WHERE continent = 'Asia';

1. **More Selects with AND/OR clauses**

use world;

SELECT \* FROM country WHERE continent = 'Africa';

SELECT count(\*) FROM country WHERE continent = 'Africa';

SELECT \* FROM country WHERE continent = 'Africa' and region = 'Western Africa';

SELECT count(\*) FROM country WHERE continent = 'Africa' and region = 'Western Africa';

SELECT \* FROM country WHERE continent = 'Africa' or region = 'Central Africa'

SELECT \* FROM country WHERE continent = 'Africa' and region = 'Western Africa'

or continent = 'Africa' and region = 'Central Africa';

1. **More Selects with IN/NOT IN clauses**

use world;

SELECT \* FROM country;

SELECT \* FROM country WHERE continent = 'Europe';

SELECT \* FROM country WHERE continent IN ('Aisa', 'Africa');

SELECT count(\*) FROM country WHERE continent IN ('Aisa', 'Africa', 'South America');

SELECT count(\*) FROM country WHERE continent IN ('Aisa', 'Africa', 'South America')

AND region NOT IN ('Caribbean', 'Southern Europe')