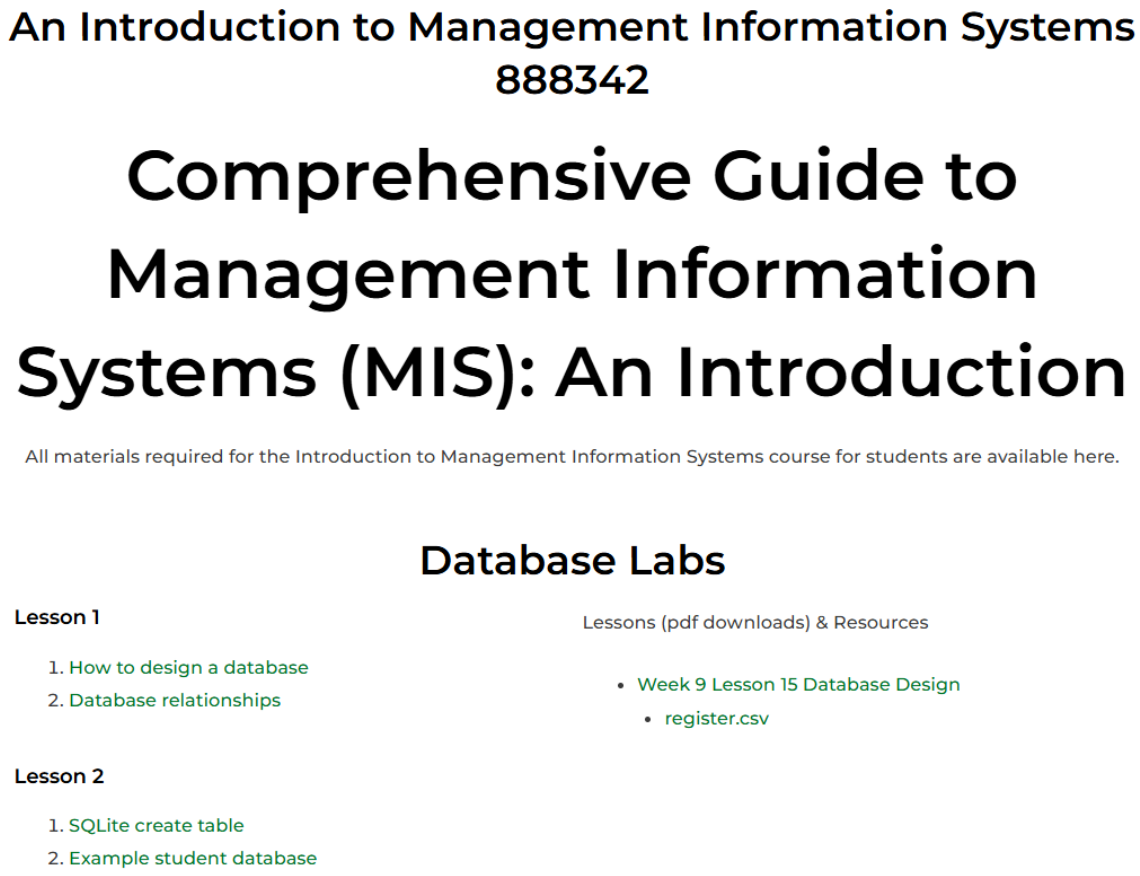
Introduction to Management Information Systems

Database Design - 2

Data Resource Management

to keep up to date & have access to data & code

https://www.alps.academy/management-information-systems/



An Introduction to Management Information Systems
888342

Comprehensive Guide to Management Information Systems (MIS): An Introduction

All materials required for the Introduction to Management Information Systems course for students are available here.

Database Labs

Lesson 1

1. How to design a database
2. Database relationships

Lesson 2

1. SQLite create table
2. Example student database

Lessons (pdf downloads) & Resources

- Week 9 Lesson 15 Database Design
  - register.csv

Relational Database Table

implement our design

1. Create Table
2. SQL Select

our task today is to

- create our database
- enter data into our database
- see the data

Preparation

1. Use your ER diagram
2. Data Types
3. Test Data

Database

1. https://sqlitebrowser.org/dl/
2. DB Browser for SQLite - Standard installer for 64-bit Windows

Database Design

table	columns	data types	notes
student	student_id	integer	primary key
student	st_firstname	varchar(50)	
lecturer	staff_id	integer	primary key
…	…		…
unit	student_id	integer	foreign key

task #1

Create Table

example create table

- create table
- columns
- data types
- *check*
- keys

table

start with one table

the following are *just* examples

create table

```
CREATE TABLE student (
    student_id
    class_no
    name
    mark
);
```

create table

```
CREATE TABLE student (
    student_id INT,
    class_no INT,
    name VARCHAR(255),
    mark INT
);
```

- create table
- columns
- data types

create table

```
CREATE TABLE student (
    student_id INT PRIMARY KEY,
    class_no INT,
    name VARCHAR(255),
    mark INT
);
```

- create table
- columns
- data types
- keys

create table

```
CREATE TABLE student (
    student_id INT PRIMARY KEY,
    class_no INT,
    name VARCHAR(255),
    mark INT CHECK (mark >= 0 AND mark <= 100)
);
```

- create table
- columns
- data types
- keys
- *check*

examples of create table

```
CREATE TABLE customer (
    customerid    float,
    firstname varchar ( 255 ),
    lastname  varchar ( 255 ),
    city     varchar ( 255 ),
    state   varchar ( 255 )
);
```

examples of create table

```
CREATE TABLE customer (
    customerid    float NOT NULL primary key,
    firstname varchar ( 255 ),
    lastname  varchar ( 255 ) NOT NULL,
    city     varchar ( 255 ),
    state   varchar ( 255 )
);
```

examples of create table

```
CREATE TABLE myemployee (
    firstname varchar ( 30 ),
    lastname varchar ( 50 ),
    title    varchar ( 30 ),
    age      number ( 3 ),
    salary number ( 10 , 2 )
);
```

examples of create table

```
CREATE TABLE myemployee (
    ID integer primary key autoincrement,
    firstname varchar ( 30 ),
    lastname varchar ( 50 ),
    title    varchar ( 30 ),
    age      number ( 3 ) check (age>=18),
    salary number ( 10 , 2 ) check (salary >= 10000)
);
```

examples of create table

```
CREATE TABLE items_ordered (
    orderid    integer,
    customerid    float,
    order_date    timestamp,
    item    varchar ( 255 ),
    quantity   float,
    price   float
);
```

examples of create table

```
CREATE TABLE items_ordered (
    orderid     integer NOT NULL unique primary key,
    customerid    float,
    order_date    timestamp,
    item    varchar ( 255 ),
    quantity   float check (quantity>=18),
    price   float check (quantity>=0),
    FOREIGN KEY (customerid) REFERENCES customers(customerid)
);
```

create the student & course tables

write your own SQL code to create a table for

1. student
2. course

try it for yourself first – (**the idea is to learn** not do)

then compare with AI

Database data entry

enter data

entering data is not important here

- we can copy & paste into sqlite

or

- use AI to create the insert statements

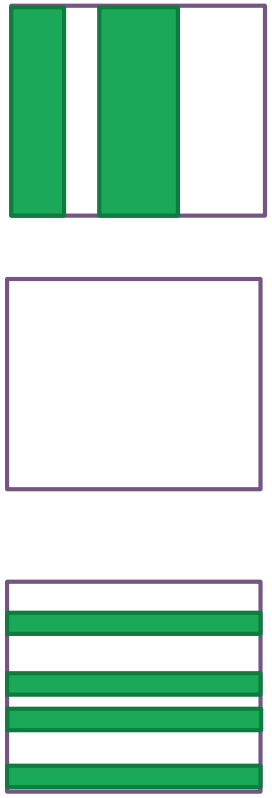
in real life we would automate & import data

Using or querying Database data

see the data

SQL Select Statements

SQL - DML



SELECT	what you want to see
FROM	from where
WHERE	which rows

SELECT *column*
FROM *tables(s)*
WHERE *age* > 22

select

```
select student_id, name
from student;
```

```
-- Select all columns from the student table
SELECT * FROM student;
-- Select specific columns from the student table
SELECT student_id, student_name FROM student;
```

select *

```
select *
from student;
```

```
-- Select all columns from the student table
SELECT * FROM student;
-- Select specific columns from the student table
SELECT student_id, student_name FROM student;
```

order by

```
select *
from student
order by student_name;
```

```
-- Select all students sorted by name in ascending order
SELECT * FROM student ORDER BY student_name ASC;
-- Select all students sorted by name in descending order
SELECT * FROM student ORDER BY student_name DESC;
```

order by

```
select *
from student
order by student_name desc;
```

```
-- Select all students sorted by name in ascending order
SELECT * FROM student ORDER BY student_name ASC;
-- Select all students sorted by name in descending order
SELECT * FROM student ORDER BY student_name DESC;
```

where =

```
select *
from student
where student_id = 652415501


-- Select students with student_id greater than 652415505
SELECT * FROM student WHERE student_id > 652415505;
-- Select courses with course_date equal to '2024-2'
SELECT * FROM course WHERE course_date = '2024-2';
```

where >

```
select *
from student
where student_id > 652415501


-- Select students with student_id greater than 652415505
SELECT * FROM student WHERE student_id > 652415505;
-- Select courses with course_date equal to '2024-2'
SELECT * FROM course WHERE course_date = '2024-2';
```

where =

```
select *
from course
where course_date = “2024-2”;


-- Select students with student_id greater than 652415505
SELECT * FROM student WHERE student_id > 652415505;
-- Select courses with course_date equal to '2024-2'
SELECT * FROM course WHERE course_date = '2024-2';
```

like – starts with

```
select student_id,name
from students
where student_id like '64%‘


-- Select students whose name starts with 'Z'
SELECT * FROM student WHERE student_name LIKE 'Z%';
-- Select courses with names containing 'Management'
SELECT * FROM course WHERE course_name LIKE '%Management%';
```

like – contains

```
select *
from course
where course_name LIKE '%Management%';


-- Select students whose name starts with 'Z'
SELECT * FROM student WHERE student_name LIKE 'Z%';
-- Select courses with names containing 'Management'
SELECT * FROM course WHERE course_name LIKE '%Management%';
```

does this work?

```
select *
from course
where course_name LIKE '%management%';


-- Select students whose name starts with 'Z'
SELECT * FROM student WHERE student_name LIKE 'Z%';
-- Select courses with names containing 'Management'
SELECT * FROM course WHERE course_name LIKE '%Management%';
```

alias

```
select name as new_name
from students


-- Rename column outputs
SELECT student_id AS ID, student_name AS Name FROM student;
```

alias

```
select first_name + “ “ + surname as fullname
from students


-- Rename column outputs
SELECT student_id AS ID, student_name AS Name FROM student;
```

arithmetic operators + - * /

```
select name, mark, mark*1.1
from students
```

later - join

```
-- Use table aliases for shorter references
SELECT s.student_id, s.student_name, r.course_code
FROM student AS s
JOIN registration AS r ON s.student_id = r.student_id;
```

multiple table

is the design ok ?

test data will tell you (keep a copy!)

consider

normalization

- address – 1 field or more
- contact details = myemail@gmail.com, 012345678
  - should be email, phone

consider

constraints – check, mark 0 to 100

insert, update , delete

look at the code for the foreign key

can you do this for student & course, how?

Thank you!

any questions?