Module Learning Outcomes

Learning Outcome	
1) DEMONSTRATE KNOWLEDGE AND CRITICAL UNDERSTANDING OF INFORMATICS AND THE UNDERPINNING THEORIES.	**Learning**
2) SELECT APPROPRIATE PRACTICES AND TOOLS TO DESIGN AN EFFECTIVE INFORMATION SYSTEM.	**Analysis** **Enquiry**
3) RETRIEVAL OF INFORMATION USING SQL.	Application

1) Porting Your ERD and Normalised Data into QSEE

Using the Employees on a Project (Normalisation Exercise 1)

A company has a number of **projects** on the go at any one time and each project will have several **employees** working on it. The employees do not necessarily stay for the full duration of the project but instead join it when they are required and leave when their particular specialist skills are no longer needed. Each employee is paid a salary and this is determined by the pay grade they are on.

UNF sample data is shown below:

Project Code	Project Description	Employee Number	Employee Firstname	Employee Surname	Pay Grade	Salary	Date Joined Project	Months Allocated To Project
A21	Allied Carpets	12	Tom	Jones	7	30000	05/05/2000	12
		56	Andrea	Murray	5	24000	05/05/2000	9
		60	Bob	Roberts	6	27000	01/07/2000	10
G02	Game	25	Jenny	Smith	5	24000	04/05/2002	`18
I11	Iceland	12	Tom	Jones	7	30000	31/12/2000	10
S03	Sainsburys	12	Tom	Jones	7	30000	01/01/2000	12
		56	Andrea	Murray	5	24000	31/05/2000	6
Z04	Zavvi	56	Andrea	Murray	5	24000	01/08/2000	12

3NF version of the database

Project (Proj_Code, Proj_Desc)

Emp_On_Project (Proj_Code, Emp_No#, Date_Joined_Proj, Months_Allocated_To_Proj)

Employee (Emp No#, Emp_fname, Emp_sname, Grade#)

Pay_Structure (Grade#, Salary#)

Some Data Types in MySQL

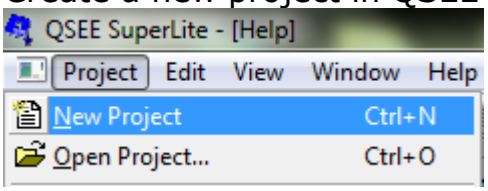
What type of data?	Type	Usage
Date	Date	MySQL displays DATE values in 'YYYY-MM-DD' format
True or False	TinyInt	Boolean – true or false[1]
-128 to 127[2]	TinyInt	Whole number
-32768 to 32767[3]	SmallInt	Whole number
-2147483648 to 2147483647[4]	Integer/ Int	Whole number
Numbers with a decimal place – such as currency	Real	Numbers with a decimal place (real is the alias for Double Precision)
A fixed number of characters	VarChar(m)	m represents the maximum length in characters

Full breakdown of data types available here:
http://dev.mysql.com/doc/refman/5.0/en/data-type-overview.html

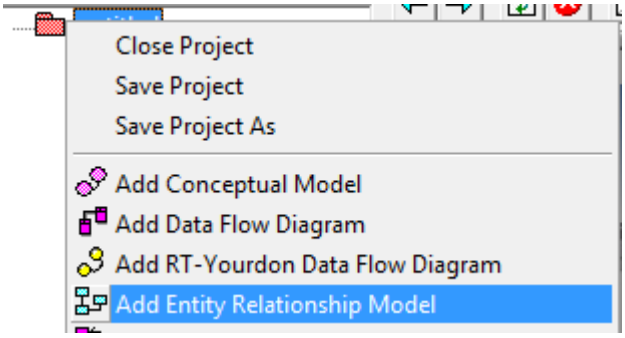
Note: To make a field unsigned (see footnotes 2 to 4 below) – you need to amend the SQL generated by QSEE before it is added to MySQL (believe me it is simpler than the alternative).

Building an ERD from the 3NF Entities and Attributes

- Create a new project in QSEE

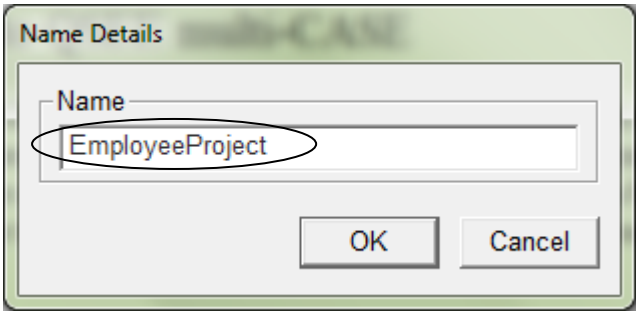


- Right click on the project title and select "**Add an Entity Relationship model**"

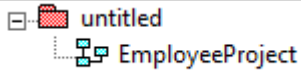


Name your project "**EmployeeProject**"

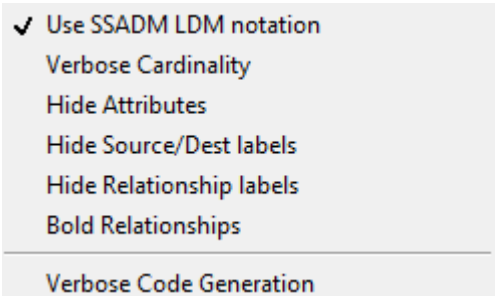
[1] Bit is the better option but not included in QSEE – you would have to change it when you had imported the SQL into MySQL
[2] 0 to 255 (unsigned)
[3] 0 to 65535 (unsigned)
[4] 0 to 4294967295 (unsigned)



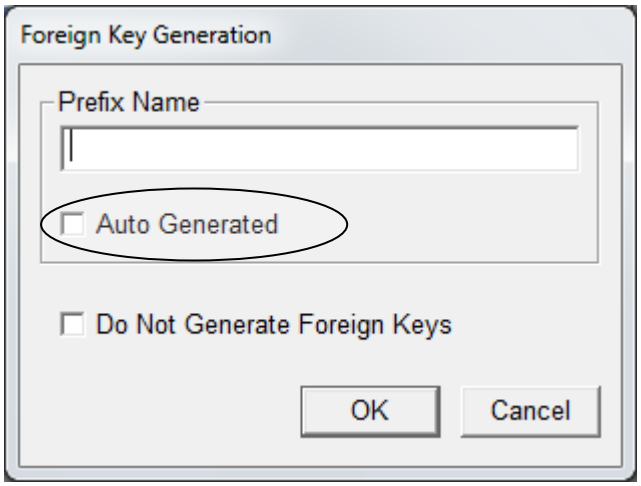
- Expand the project title so you can see your ERD:



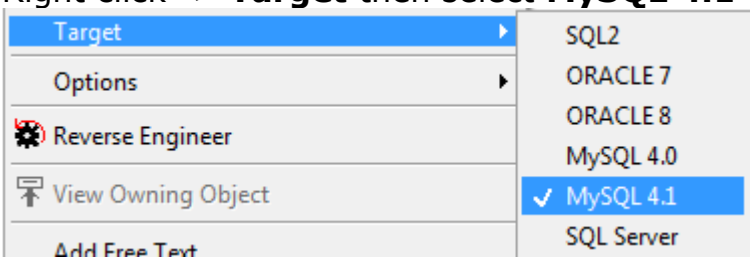
- Right click on the ERD name and select **Options** then change the following:
  - Use SSADM LDM Notation [Check]
  - Verbose Cardinality [Uncheck]
  - Verbose Code Generation [Uncheck]



- Right click on the ERD name and select **Options** (again) then click on Foreign Key Generation (global) and change the following:
  - Auto Generated [Uncheck]



- Right click -> **Target** then select **MySQL 4.1**



Note: You are about to create the following ERD diagram – it is included in full here so that as you go you can lay it out correctly.

3NF version of the database (included again so that you can compare it with the QSEE ERD below)

Project (Proj_Code, Proj_Desc)
Emp_On_Project (Proj_Code, Emp_No#, Date_Joined_Proj, Months_Allocated_To_Proj)
Employee (Emp_No#, Emp_fname, Emp_sname, Grade#)
Pay_Structure (Grade#, Salary#)

Note: You do not need to add fields that are foreign keys. QSEE assumes you want them by simply adding a relationship between two entities. Foreign keys are not shown in QSEE though they do appear when the data is exported to MySQL.

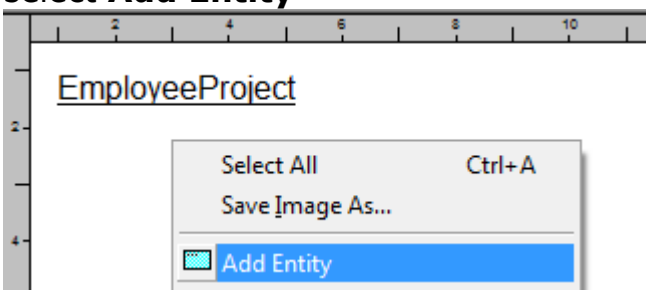
Note: The Emp_On_Project entity has a dotted outline – this is because it has no primary key assigned at this point (because the primary key in this table is a composite key based on Emp_No and Proj_Code – and they are foreign keys from their respective tables).

Project

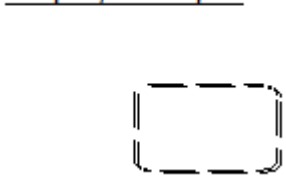
Project (Proj_Code, Proj_Desc)

What is the entity name? For each attribute: - Is the attribute(s) a primary key? - What data type does it need to be – *look at the sample data*? - For fields other than the primary key - can the field be blank?

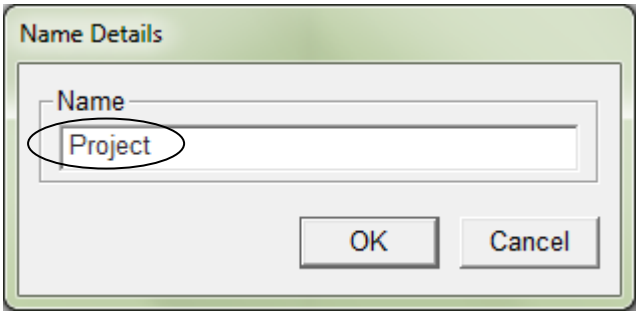
- **Right click** in the large area below the ERD diagram name (as shown) and select **Add Entity**



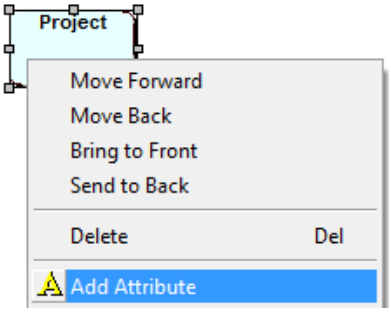
- You will then see a dotted rectangle which will become your entity



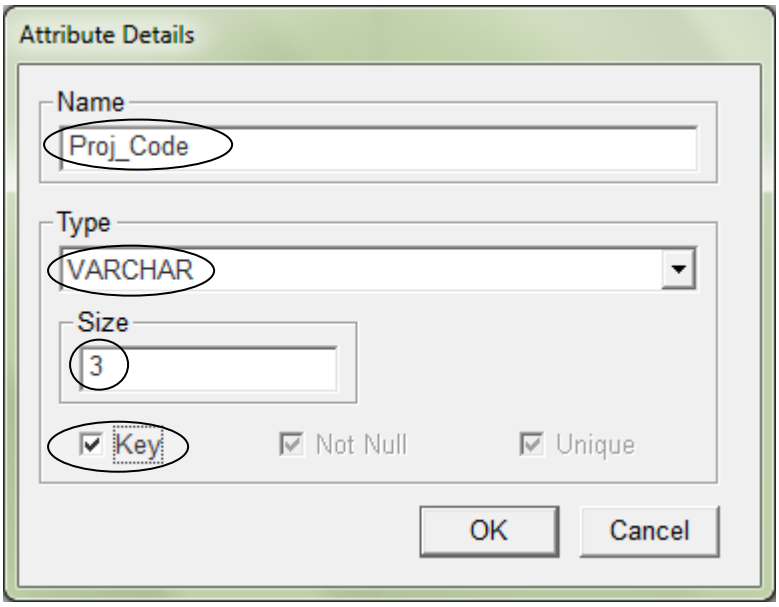
- Left click when you are satisfied with its position on the ERD diagram. You will then be asked to enter the name of the entity. Call it "Project"



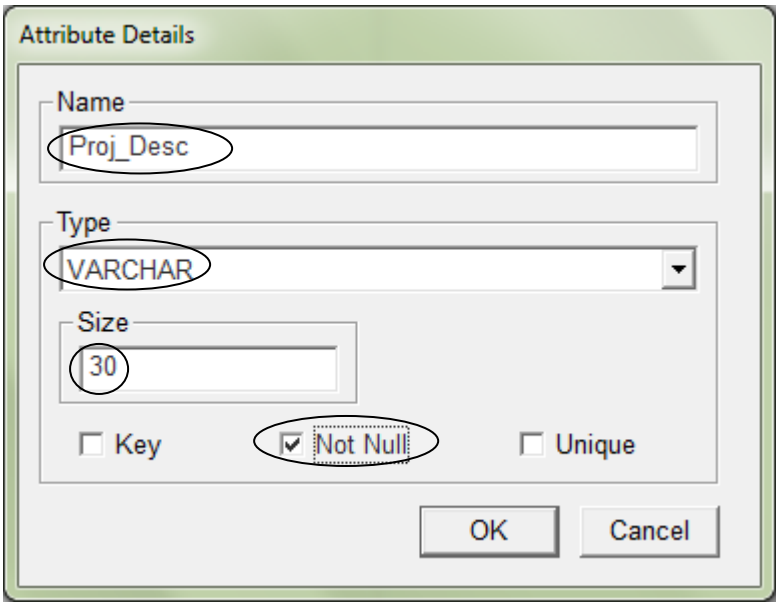
- Right click on Project then select **Add Attribute**



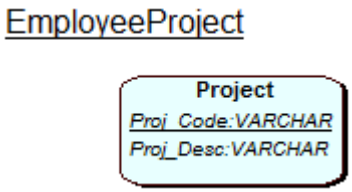
- Enter the following attribute details as highlighted below:



- Right click on Project again and select **Add Attribute**



Your diagram should now look like this (showing the two attributes including primary key):



Pay Structure

Pay_Structure (Grade#, Salary#)

What is the entity name? For each attribute: - Is the attribute(s) a primary key? - What data type does it need to be – *look at the sample data*? - For fields other than the primary key - can the field be blank?

- **Right click** in the large area below the ERD diagram name (as shown) and select **Add Entity**
- You will then see a dotted rectangle which will become your entity. Left click when you are satisfied with its position on the ERD diagram.
- You will then be asked to enter the name of the entity. Call it "**Pay_Structure**"
- Right click on Pay_Structure then select **Add Attribute**
    - Name: Grade
    - Type: Tiny Int
    - Key [Check]
- Right click on Pay_Structure then select **Add Attribute**
    - Name: Salary
    - Type: Real
    - Not Null [Check]

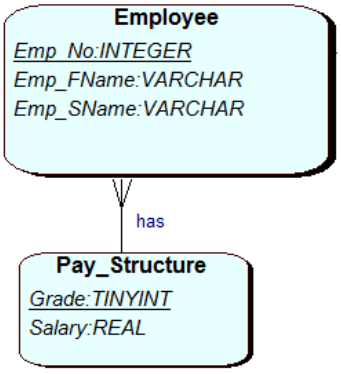
Employee

Employee (Emp No#, Emp_fname, Emp_sname, Grade#)

What is the entity name? For each attribute: - Is the attribute(s) a primary key? - What data type does it need to be – *look at the sample data*? - For fields other than the primary key - can the field be blank?

- **Right click** in the large area below the ERD diagram name and select **Add Entity**
- You will then see a dotted rectangle which will become your entity. Left click when you are satisfied with its position on the ERD diagram.
- You will then be asked to enter the name of the entity. Call it "**Employee**"
- Right click on Employee then select **Add Attribute**
    - Name: **Emp_No**
    - Type: Integer (ideally SmallInt)
    - Key [Check]
- Right click on Employee then select **Add Attribute**
    - Name: **Emp_fname**
    - Type: VarChar -> Size 30
    - Not Null [Check]
- Right click on Employee then select **Add Attribute**
    - Name: **Emp_sname**
    - Type: VarChar -> Size 30
    - Not Null [Check]
- **DO NOT ADD a Grade attribute** – instead do the following (which adds the Grade field to the Employee entity though you won't see it in QSEE - but it show up in MySQL):
    - Right Click on Pay_Structure -> **Add Relationship**
      -> Click on Employee
      Relationship "has"
      Cardinality 1:m

This part of the ERD diagram should now look like this:



Employee on Project

Emp_On_Project (Proj_Code, Emp_No#, Date_Joined_Proj, Months_Allocated_To_Proj)

What is the entity name? For each attribute: - Is the attribute(s) a primary key? - What data type does it need to be – *look at the sample data*? - For fields other than the primary key - can the field be blank?

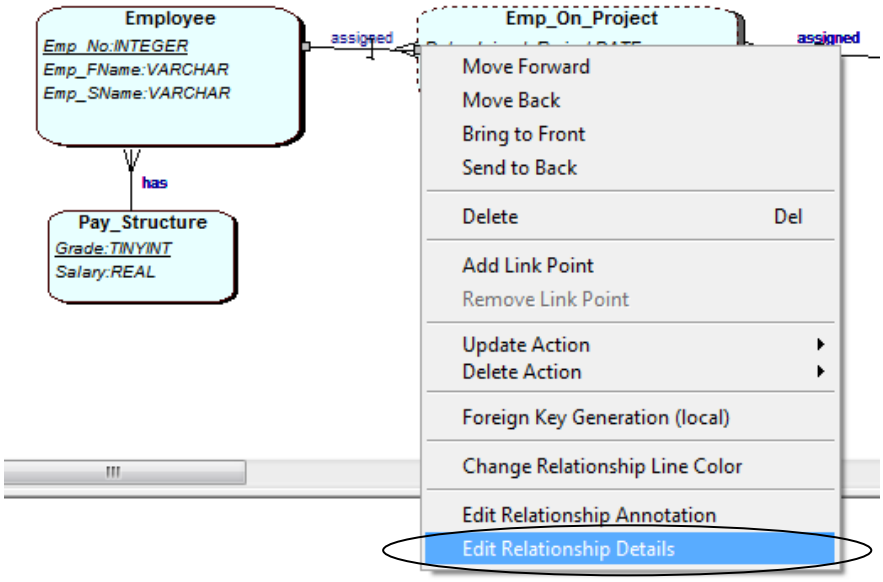
- **Right click** in the large area below the ERD diagram name and select **Add Entity**
- You will then see a dotted rectangle which will become your entity. Left click when you are satisfied with its position on the ERD diagram.
- You will then be asked to enter the name of the entity. Call it "**Emp_On_Project**"
- **DO NOT ADD the Proj_Code attribute** – instead do the following (which adds the Proj_Code field to Emp_On_Project but it show up in MySQL):
    - Right Click on Emp_On_Project -> Add Relationship
      -> Click on Project
      Cardinality m:1
- **DO NOT ADD the Emp_No attribute** – instead do the following (which adds the Emp_No field to Emp_On_Project but it show up in MySQL):
    - Right Click on Emp_On_Project -> Add Relationship
      -> Click on Employee
      Cardinality m:1
- Right click on Emp_On_Project then select **Add Attribute**
    - Name: Date_Joined_Proj
    - Type: Date
    - Not Null [Check]
- Right click on Emp_On_Project then select **Add Attribute**
    - Name: Months_Allocated_To_Proj
    - Type: TinyInt

- Not Null [Check]

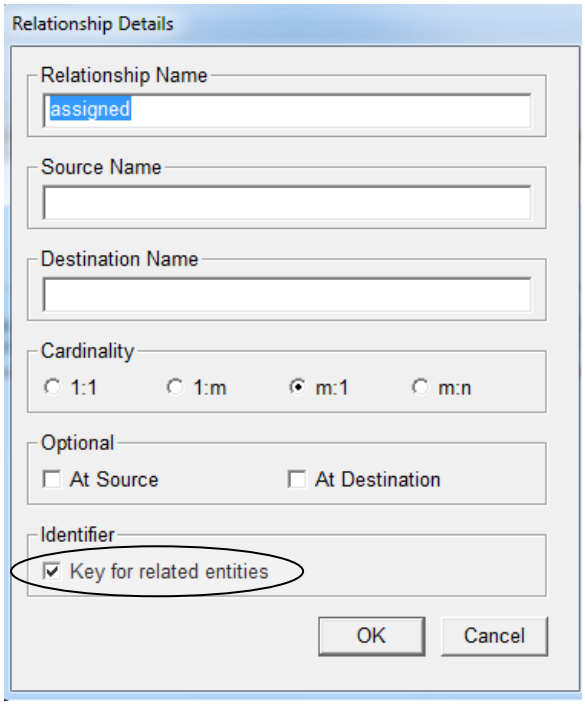
<u>To ensure that the Emp_On_Project has a composite primary key</u>

Remember Emp_No and Proj_Code do not appear in Emp_On_Project as they are foreign keys (though they do exist). To make them a composite/joint primary key we need to do the following:

Right click on the relationship between Employee and Emp_On_Project and select **Edit Relationship Details**.



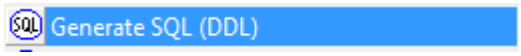
Then ensure that the Identifier "Key for related entities" check box is selected.



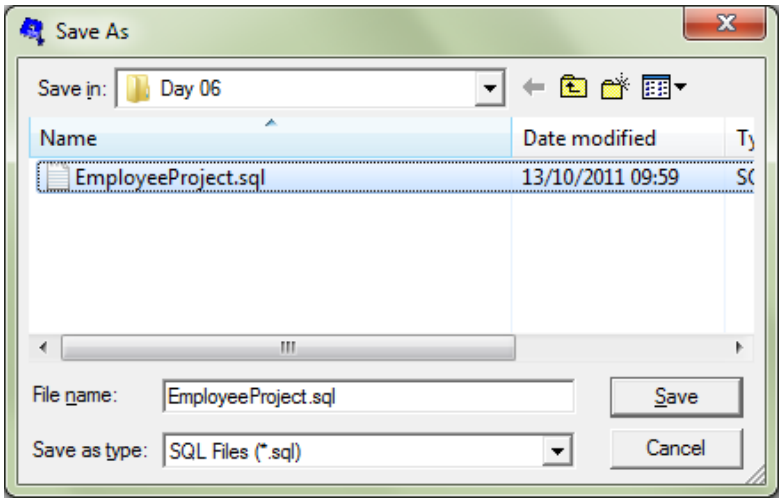
Do the same thing for the relationship between Emp_On_Project and Proj_Code.

2) Exporting your QSEE model as SQL

In the left hand window right click on EmployeeProject in the left hand window then select "**Generate SQL (DDL)**"



You will then be presented with the following dialog box.



Note: The file created is not perfect – i.e. it will not load as it is in MySQL!
You need to open it in notepad then edit the file and replace all "---" with "-- " and replace all "TYPE=INNODB;" with ";"
DO NOT INCLUDE THE QUOTATION MARKS WHEN REPLACING THE TEXT

I am including the ERD again here so you can compare it with the SQL statements below:

Project SQL

```
-- Create a Database table to represent the "Project" entity.
CREATE TABLE Project(
        Proj_Code  VARCHAR(3) NOT NULL,
        Proj_Desc  VARCHAR(30) NOT NULL,

        -- Specify the PRIMARY KEY constraint for table "Project".
        -- This indicates which attribute(s) uniquely identify each row of data.
        CONSTRAINT      pk_Project PRIMARY KEY (Proj_Code)
);
```

Pay_Structure SQL

```
-- Create a Database table to represent the "Pay_Structure" entity.
CREATE TABLE Pay_Structure(
        Grade           TINYINT NOT NULL,
        Salary          REAL NOT NULL,
        -- Specify the PRIMARY KEY constraint for table "Pay_Structure".
        -- This indicates which attribute(s) uniquely identify each row of data.
        CONSTRAINT      pk_Pay_Structure PRIMARY KEY (Grade)
);
```

Employee SQL

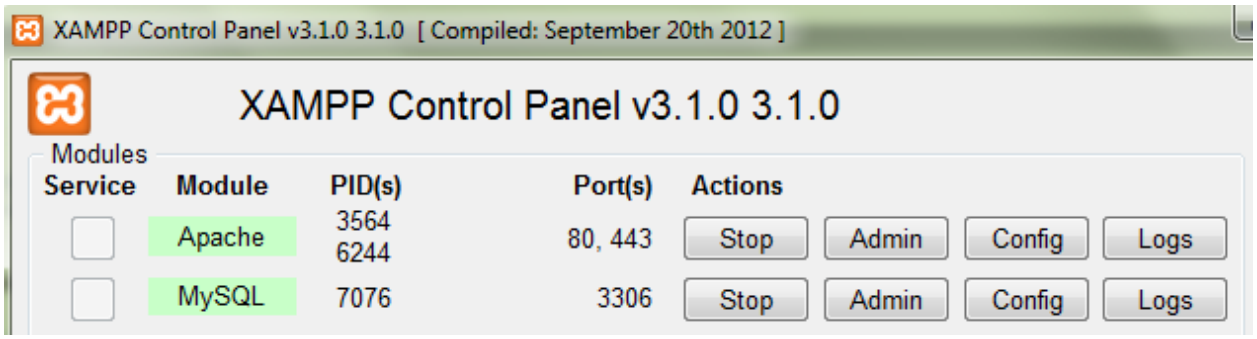
```
-- Create a Database table to represent the "Employee" entity.
CREATE TABLE Employee(
        Emp_No                  INTEGER NOT NULL,
        Emp_FName               VARCHAR(30) NOT NULL,
        Emp_SName               VARCHAR(30) NOT NULL,
        Grade                   TINYINT NOT NULL,
        -- Specify the PRIMARY KEY constraint for table "Employee".
        -- This indicates which attribute(s) uniquely identify each row of data.
        CONSTRAINT      pk_Employee PRIMARY KEY (Emp_No)
);
ALTER TABLE Employee ADD INDEX (Grade), ADD CONSTRAINT
fk1_Employee_to_Pay_Structure FOREIGN KEY(Grade) REFERENCES
Pay_Structure(Grade) ON DELETE RESTRICT ON UPDATE RESTRICT;
```

Emp_On_Project SQL (where is the key?)

```
-- Create a Database table to represent the "Emp_On_Project" entity.
CREATE TABLE Emp_On_Project(
        Date_Joined_Project             DATE NOT NULL,
        Months_Allocated_To_Project     TINYINT NOT NULL,
        Proj_Code                       VARCHAR(3) NOT NULL,
        Emp_No                          INTEGER NOT NULL
        CONSTRAINT      pk_Emp_On_Project PRIMARY KEY (Proj_Code,Emp_No)
);
ALTER TABLE Emp_On_Project ADD INDEX (Proj_Code), ADD CONSTRAINT
fk1_Emp_On_Project_to_Project FOREIGN KEY(Proj_Code) REFERENCES
Project(Proj_Code) ON DELETE RESTRICT ON UPDATE RESTRICT;
ALTER TABLE Emp_On_Project ADD INDEX (Emp_No), ADD CONSTRAINT
fk2_Emp_On_Project_to_Employee FOREIGN KEY(Emp_No) REFERENCES
Employee(Emp_No) ON DELETE RESTRICT ON UPDATE RESTRICT;
```

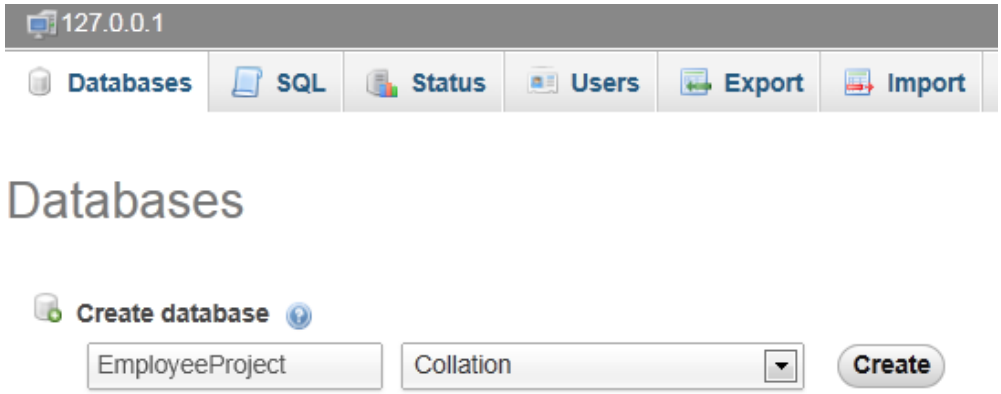
3) Importing your SQL into XAMPP

Launch XAMPP – then click the start buttons next to both the Apache and MySql modules



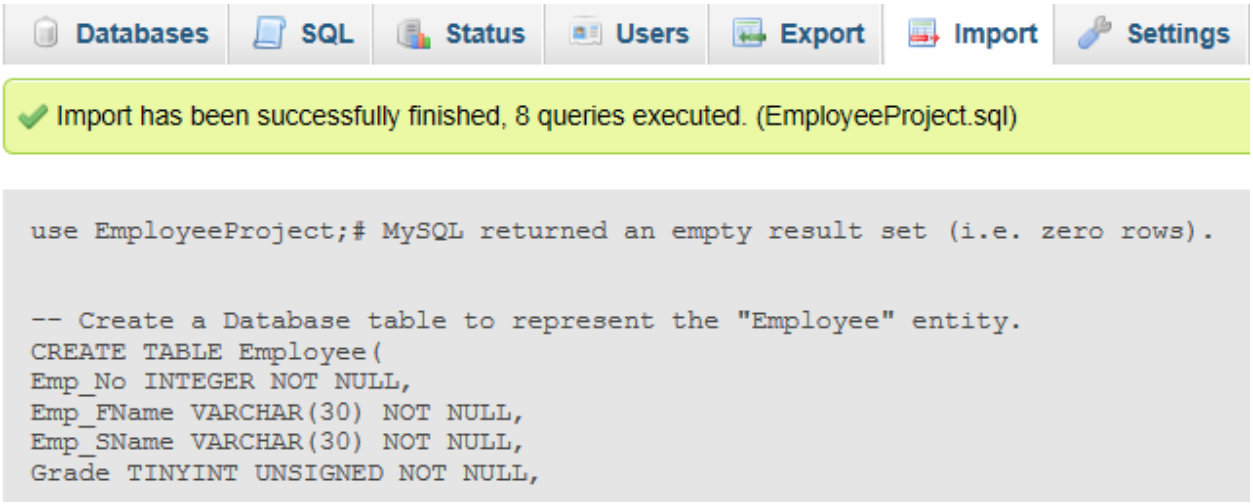
Next, click the Admin button next to MySql (if that fails to do anything enter the following into your browser http://localhost/phpmyadmin/)

Select the **Databases** tab then enter **EmployeeProject** (same name as you specified in QSEE superlite) then click the Create button

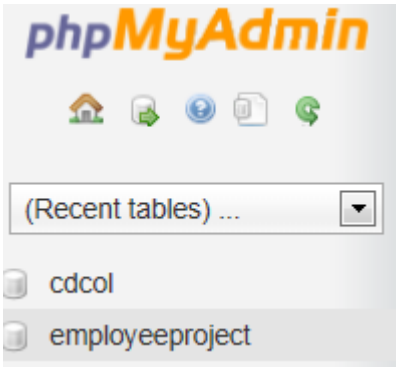


Click on the Import tab then Browse and select the sql file you exported from QSEE – then click the Go button

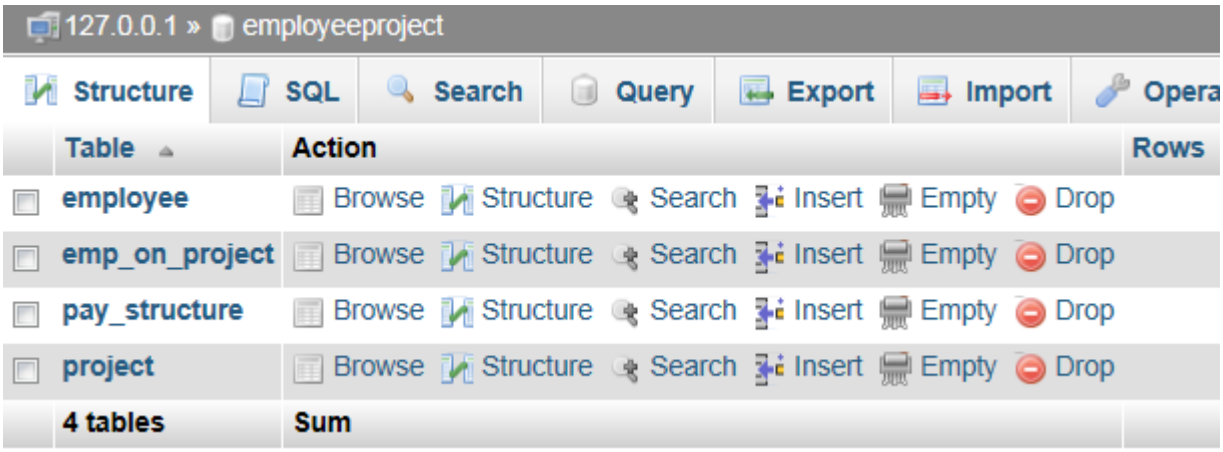
If all goes well then you will see the following screen:



Select **employeeproject** from the left-hand window:



You will now see the following summary:



You can examine each of your tables by selecting a table name. For example clicking on Project:

Project (primary key is underlined)

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**Proj_Code**	varchar(3)	latin1_swedish_ci		No	*None*	
2	**Proj_Desc**	varchar(30)	latin1_swedish_ci		No	*None*	

Pay_Structure (primary key is underlined)

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**Grade**	tinyint(3)		UNSIGNED	No	*None*	
2	**Salary**	double			No	*None*	

Employee (primary key is underlined)

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**Emp_No**	int(11)			No	*None*	
2	**Emp_FName**	varchar(30)	latin1_swedish_ci		No	*None*	
3	**Emp_SName**	varchar(30)	latin1_swedish_ci		No	*None*	
4	**Grade**	tinyint(3)		UNSIGNED	No	*None*	

Emp_On_Project (primary keys are underlined)

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**Date_Joined_Project**	date			No	*None*	
2	**Months_Allocated_To_Project**	tinyint(4)			No	*None*	
3	**Proj_Code**	varchar(3)	latin1_swedish_ci		No	*None*	
4	**Emp_No**	int(11)			No	*None*	

If you look below this table you will see a link called + Indexes . Click on this and you will see any indexes on this table.

- Indexes

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	**PRIMARY**	BTREE	Yes	No	Emp_No	0	A	No	
Edit Drop	**Grade**	BTREE	No	No	Grade	0	A	No	

You can see that Grade has an index (both primary and foreign key fields have indexes).

Click on Relation view

Print view Relation view Propose table structure Track table

and you can see that Grade is a foreign key to Grade in Pay_Structure.

Relations

Column	Internal relation	Foreign key constraint (INNODB)
Emp_No		
Emp_FName		No index defined!
Emp_SName		No index defined!
Grade		`employeeproject`.`pay_structure`.`Grade` ON DELETE RESTRICT ON UPDATE RESTRICT