****

**Relational Database**

**Assignment#2**

**Assignment Setup**

This assignment requires the use of MySQL Workbench, along with MySQL Server.

**Assignment Setup**

This assignment is the continuation of Assignment#1 including all the new features taught in the Relational Database course. You can use the existing database or create a new one incorporating the requirements as stated below.

**Assignment Requirements.**

1. Use the Database from Assignment#1 or create a new with the same requirements. (5 points)
2. Apply new Constraints int the form Foreign Keys modifying the existing ones if necessary (15 points).
3. Create at least 4 Junction Tables using existing tables (10 points).
4. Create at least 4 Composite Keys (10 points).
5. All the tables should be Normalized based on 1st, 2nd and 3rf Normal Forms (30 points).
6. Select any type of data from any tables applying (15 points):
   1. A Cross Join.
   2. An Equijoin Inner Join.
   3. A Non-Equijoin Inner Join.
   4. A Natural Join.

Write a brief essay describing (15 points):

1. The relationships between your tables (e.g. one-to-one) and why you made these choices.
2. How and why each one of the tables is normalized.

Submission: A document that describes your tables and screenshots from the EER Schema, clearly displaying the Constraints and the Relationships among the tables..

Part 1: Describe your tables: Refer the names of each table, its constraints and its relationships.

e.g.

customers: primary key is the column 🡪 customerid, foreign key is the column 🡪 customerid referencing the column customerid in the sales table. The type of the relationship is many-to-many.

branch: primary key is the column 🡪 **branch\_id**,

foreign key is the column 🡪 **branch\_address\_id**

referencing the column **branch\_address\_id** in the **branch\_address** table.

The type of the relationship is one-to-one.

employees: primary key is the column 🡪 **employees\_id**,

foreign key is the column 🡪 **work\_type\_id, spouse\_id**

referencing the column **work \_type\_id** in the **work \_type** table.

The type of the relationship is one-to-many.

referencing the column **spouse\_id** in the **spouse** table.

The type of the relationship is one-to-one.

order\_detail: primary key is the column 🡪 **order\_detail\_id**,

foreign key is the column 🡪 **payment\_method\_id, shipping\_address\_id**

referencing the column **payment\_method\_id** in the **payment\_method** table.

The type of the relationship is one-to-many.

referencing the column **shipping\_address\_id** in the **shipping\_address** table.

The type of the relationship is one-to-one.

payment\_info: primary key is the column 🡪 **payment\_info\_id**,

foreign key is the column 🡪 **payment\_method\_id**

referencing the column **payment\_method\_id** in the **payment\_method** table.

The type of the relationship is one-to-many.

order\_detail <-> Insurance\_info (junction table : **order\_detail\_insurance\_info)** :

primary key is the column 🡪 **order\_detail\_id**, **insurance\_info\_id**

foreign key is the column 🡪 **order\_detail\_id, insurance\_info\_id**

referencing the column **order\_detail\_id** in the **order\_detail** table.

referencing the column **insurance\_info\_id** in the **insurance\_info** table.

The type of the relationship is many-to-many.

order\_detail <-> bank\_info(junction table : **order\_bank\_junction\_table)** :

primary key is the column 🡪 **order\_detail\_id**, **bank\_info\_id**

foreign key is the column 🡪 **order\_detail\_id, bank\_info\_id**

referencing the column **order\_detail\_id** in the **order\_detail** table.

referencing the column **bank\_info\_id** in the **bank\_info** table.

The type of the relationship is many-to-many.

shipping\_method: primary key is the column 🡪 **shipping\_method\_id**,

foreign key is the column 🡪 **shipping\_type\_detail\_id**

referencing the column **shipping\_type\_detail\_id** in the **shipping\_type\_detail** table.

The type of the relationship is one-to-many.

bank\_info: primary key is the column 🡪 **bank\_info\_id**,

foreign key is the column 🡪 **bank\_type\_detail\_id, account\_name**

referencing the column **bank\_type\_detail\_id** in the **bank\_type\_detail** table.

The type of the relationship is one-to-many.

referencing the column **account\_name** in the **account\_name\_detail** table.

The type of the relationship is one-to- many.

buyer: primary key is the column 🡪 **buyer\_id**,

foreign key is the column 🡪 **product\_type\_detail\_id**

referencing the column **product\_type\_detail\_id**in the **product\_type\_detail** table.

The type of the relationship is one-to-many.

Insurance\_info <-> plan\_name\_detail(junction table : **order\_bank\_junction\_table)**:

primary key is the column 🡪 **buyer\_id**,

foreign key is the column 🡪 **product\_type\_detail\_id**

referencing the column **product\_type\_detail\_id**in the **product\_type\_detail** table.

The type of the relationship is one-to-many.

order\_detail <-> bank\_info(junction table : **insurance\_plan\_junction\_table)** :

primary key is the column 🡪 **insurance\_info\_id**, **plan\_name\_detail\_id**

foreign key is the column 🡪 **insurance\_info\_id, plan\_name\_detail\_id**

referencing the column **insurance\_info\_id** in the **insurance\_info** table.

referencing the column **plan\_name\_detail\_id** in the **plan\_name\_detail** table.

The type of the relationship is many-to-many.

order\_detail <-> buyer (junction table : **order\_detail\_buyer\_junction\_table)** :

primary key is the column 🡪 **order\_detail\_id**, **buyer\_id**

foreign key is the column 🡪 **order\_detail\_id, buyer\_id**

referencing the column **order\_detail\_id**in the **order\_detail**table.

referencing the column **buyer\_id**in the **buyer** table.

The type of the relationship is many-to-many.

order\_detail <-> seller(junction table : **order\_detail\_seller\_junction\_table)** :

primary key is the column 🡪 **order\_detail\_id**, **seller\_id**

foreign key is the column 🡪 **order\_detail\_id, seller \_id**

referencing the column **order\_detail\_id**in the **order\_detail**table.

referencing the column **seller \_id**in the **seller** table.

The type of the relationship is many-to-many.

Part 2: Paste the screenshots of the entire screen, displaying your EER.

Payment\_info <-> buyer (junction table : **payment\_info\_buyer\_junction\_table)** :

primary key is the column 🡪 **payment\_info\_id**, **buyer\_id**

foreign key is the column 🡪 **payment\_info\_id, buyer\_id**

referencing the column **payment\_info\_id** the **payment\_info** table.

referencing the column **buyer\_id**in the **buyer** table.

The type of the relationship is many-to-many.

product\_export : primary key is the column 🡪 **product\_export\_id**,

foreign key is the column 🡪 **delivery\_method\_id**

referencing the column **delivery\_method\_id** in the **delivery\_method** table.

The type of the relationship is one-to-many.

product\_import : primary key is the column 🡪 **product\_import\_id**,

foreign key is the column 🡪 **delivery\_method\_id**

referencing the column **delivery\_method\_id** in the **delivery\_method** table.

The type of the relationship is one-to-many.

product\_national : primary key is the column 🡪 **product\_national\_id**,

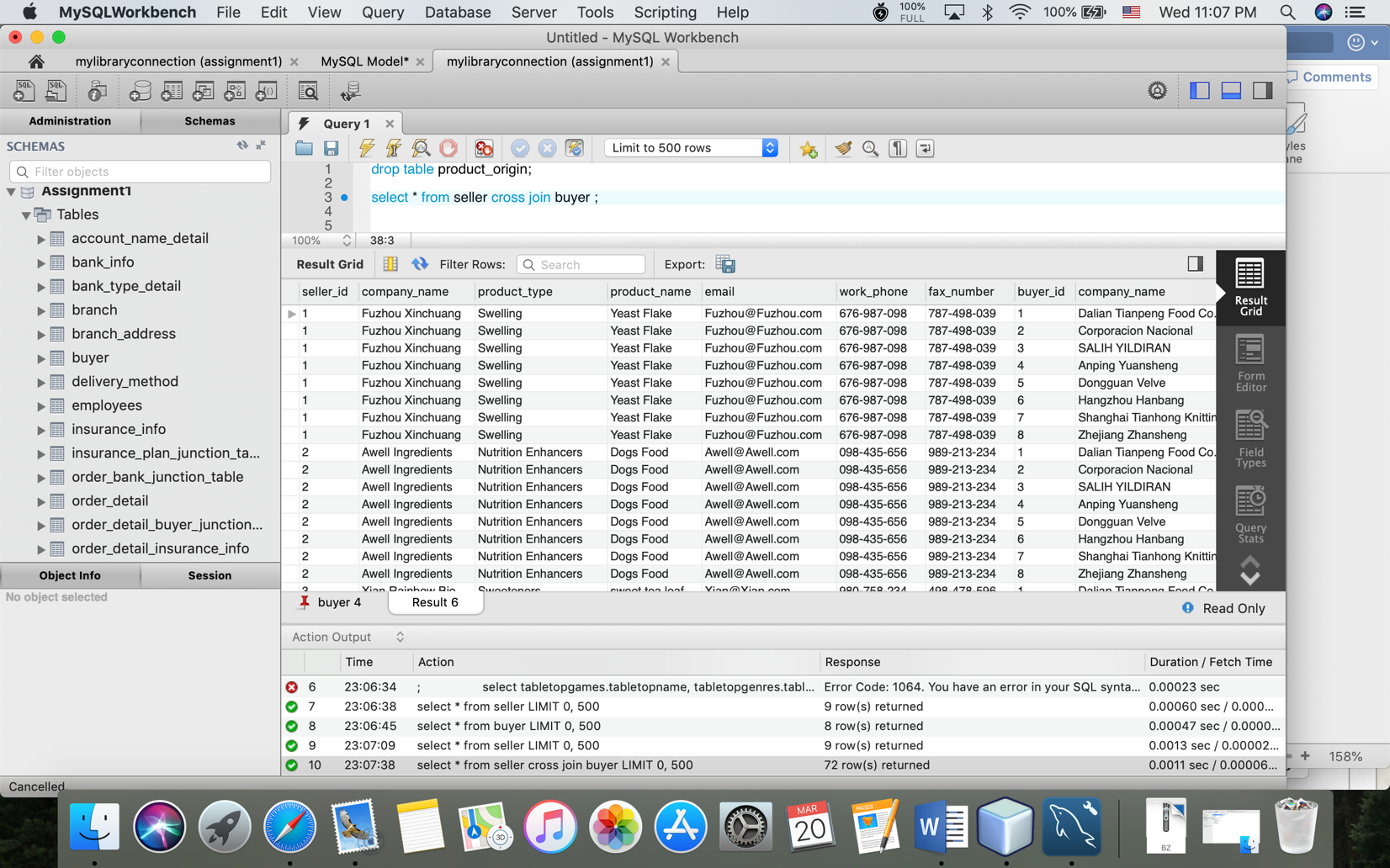
foreign key is the column 🡪 **delivery\_method\_id**

referencing the column **delivery\_method\_id** in the **delivery\_method** table.

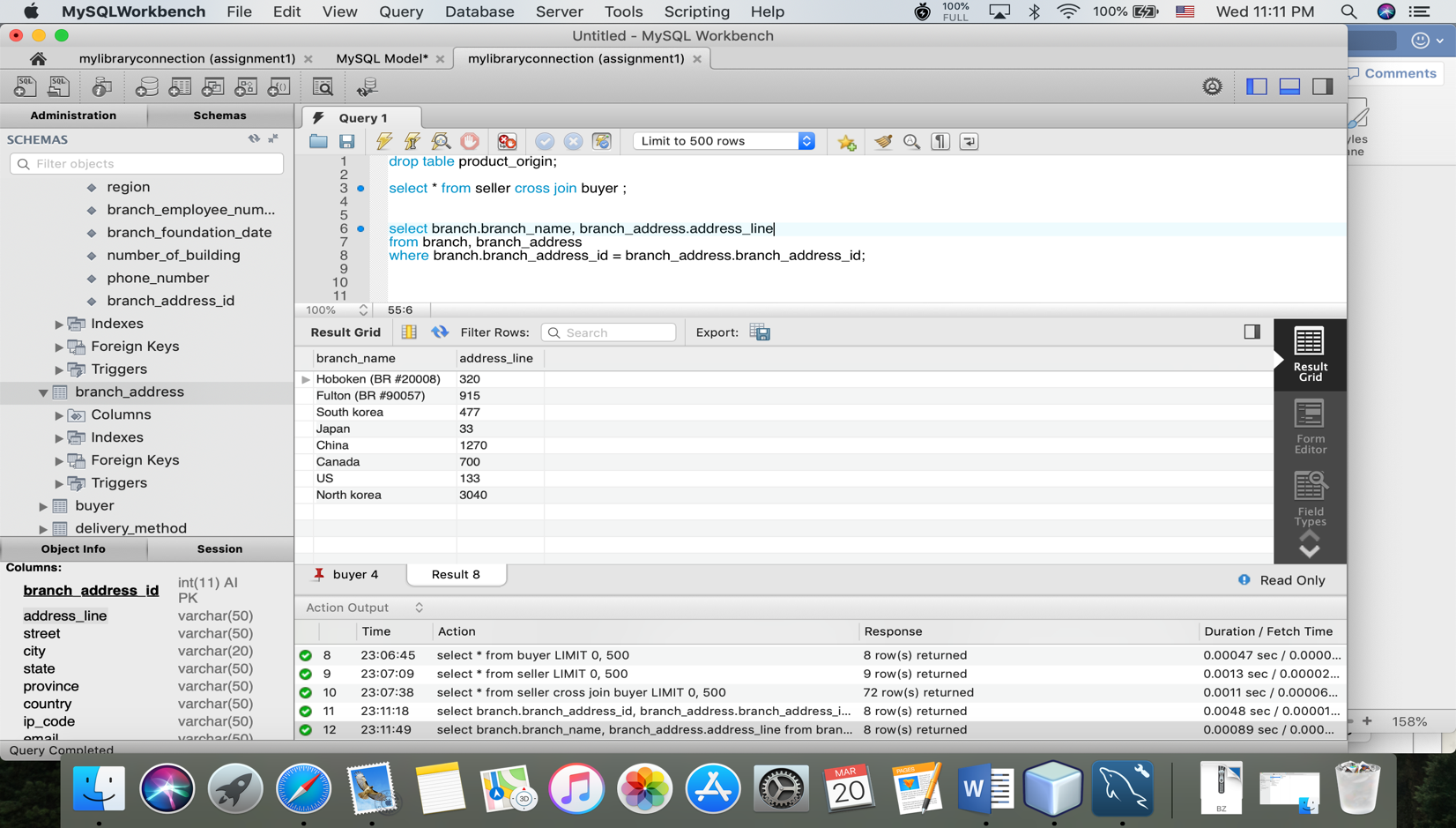
The type of the relationship is one-to-many.



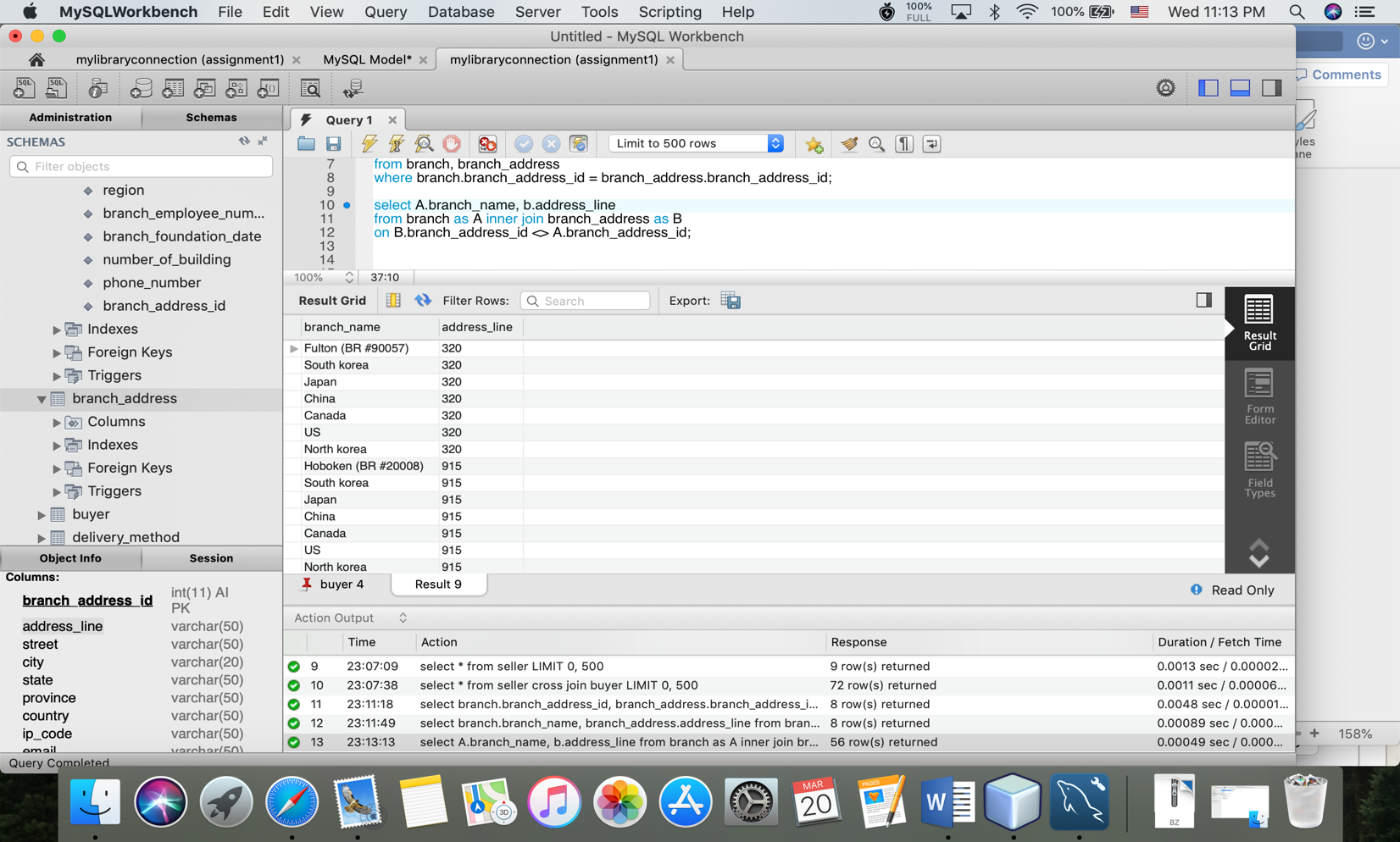
A Cross Join



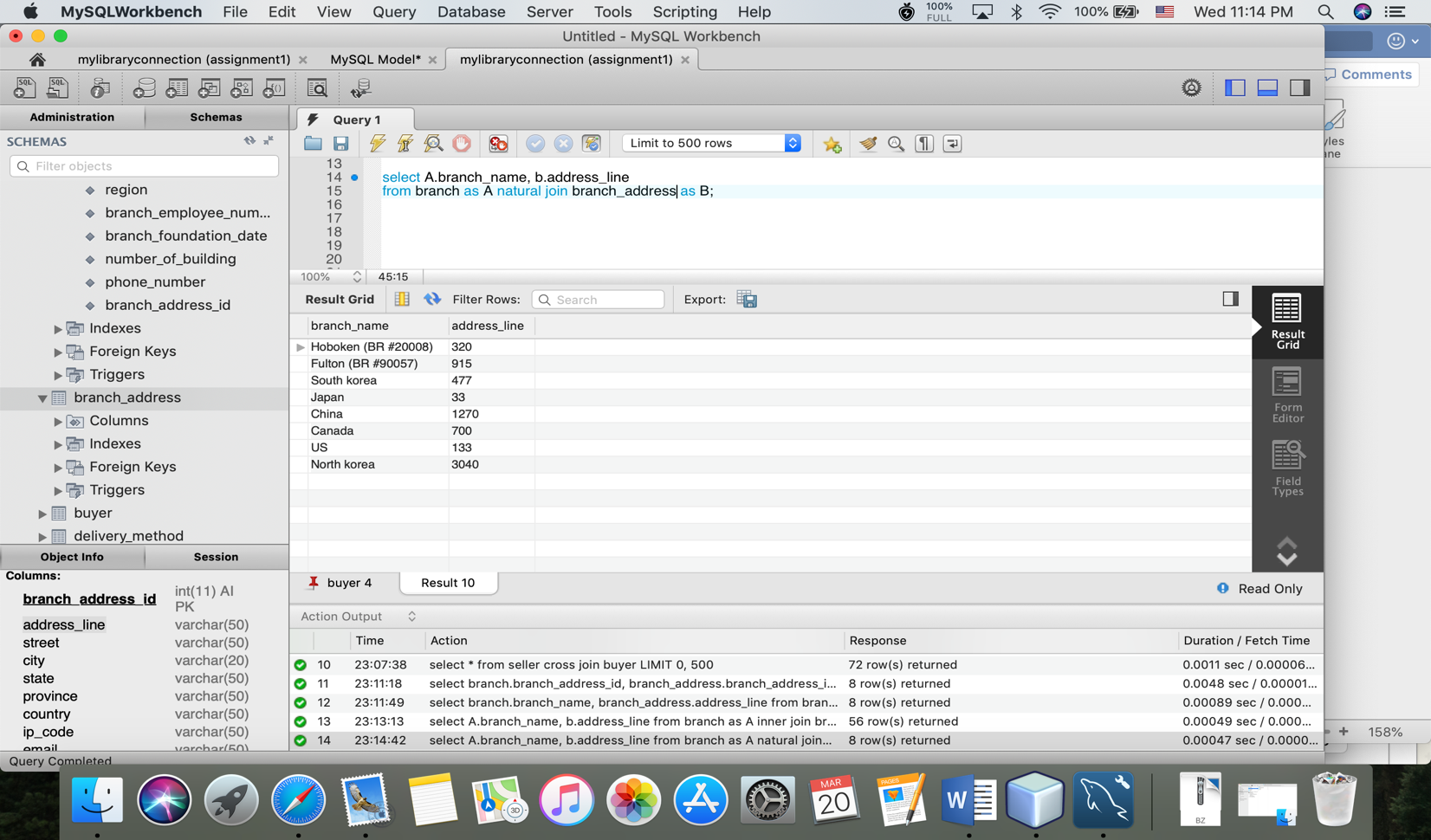
An Equijoin Inner Join



A Non-Equijoin Inner Join



A Natural Join



Write a brief essay describing (15 points):

1. The relationships between your tables (e.g. one-to-one) and why you made these choices.
2. How and why each one of the tables is normalized.

The relationships between tables:

1. One to one relationship – one record in one table is related with only one record in another table. So, two tables have only one relationship each other. The reason why I chose this relationship is that I wanted to put data in one table, but the data had many data inside. So, I made it normalize and I put the data in another table. After that, I connected two table using foreign key.
2. One to many relationship – one record in one table is related with many records in other table. The reason why I chose this relationship is that One table has only one record which has many repetitive records in another table. So, I created two tables and I connected two tables using foreign key.
3. Many to many relationship - many records in one table is related with many records in other table. The reason why I chose this relationship is that one record in one table has many records in another table and vice versa. I created three tables and two tables are related through Junction table.

Normalization: First of all, I look into my tables and, I checked my tables whether they have multiple data stored. If the data in tables are big, I separated them into single data and made other tables. And then, I defined one of the columns in the table as primary key (which I could define many primary keys in one table but in the case of one to one relationship and one to many relationships, we need only one primary key in the table).

Second, when I make one to one table, I connected two tables with one primary key as foreign key. And one to many relationship tables are almost same but the table in one relationship can’t use other data which are not included in the table with many relationships.

Many to many relationship tables are also similar but each table has to have the data included in each table.