Alejandro Sanchez

BTE423

Final Project – Case Study: Airport Information System

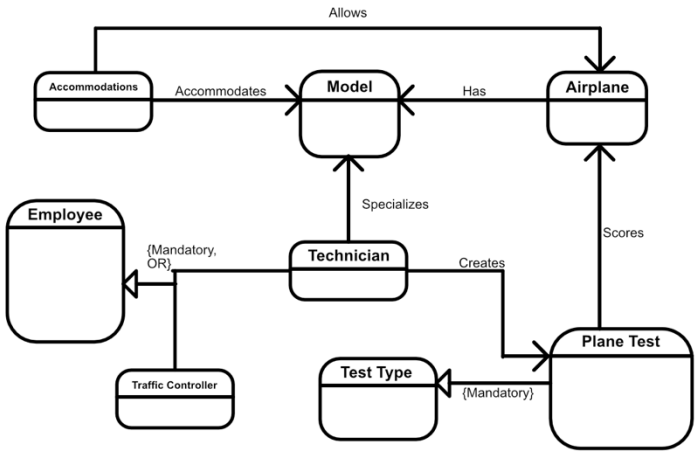
Construction of the Conceptual Data Model

The first step is to understand the layout of the required entities and how they relate to one another.

1. From the information available, the following basic entities can be interpreted:
    a. Employee
    b. Test
    c. Airport Accommodations
    d. Airplane
2. Within these entities, some attributes can be designated respectively as the description of requirements
    a. Employee
        i. Name
        ii. Social Security Number
        iii. Identification Number (added as a measure of security to not rely on SSN for other information)
        iv. Address
        v. Phone Number
        vi. Salary
        vii. Union Name
        viii. Position
    b. Test
        i. Test Number
        ii. Name
        iii. Maximum Possible Score
        iv. Plane Registration Number
        v. Technician Identification Number
        vi. Date
        vii. Hours Spent on Test
        viii. Score Received
    c. Airport Accommodations
        i. Model Number of allowed models
    d. Airplane
        i. Registration Number
        ii. Model Number
        iii. Model Capacity
        iv. Model Weight
3. Within these entities, some may be specialized to conform to the requirements of the system at hand:
    a. Employee – can be derived into Technicians and Traffic Controllers through the Position attribute.

        i. Technicians have an added attribute of Specialization
        ii. Traffic Controllers have an added attribute of Exam Date
4. Ideally, some of the attributes can be separated into other entities and then referenced, in order to limit the amount of attributes per entity and allow changes to be made if necessary (for example adjusting the weight of a model)
    a. Employee
        i. Name
        ii. Social Security Number
        iii. Identification Number (added as a measure of security to not rely on SSN for other information)
        iv. Address
        v. Phone Number
        vi. Salary
        vii. Union Number
        viii. Position
    b. Technician
        i. Identification Number
        ii. Specialization
    c. Traffic Controller
        i. Identification Number
        ii. Exam Date
    d. Test Type
        i. Test Number
        ii. Name
        iii. Maximum Possible Score
    e. Plane Test
        i. Test Number
        ii. Plane Registration Number
        iii. Technician Identification Number
        iv. Date
        v. Hours Spent on Test
        vi. Score Received
    f. Airport Accommodations
        i. Model Number of allowed models
    g. Airplane
        i. Registration Number
        ii. Model Number
    h. Model
        i. Model Number
        ii. Model Capacity
        iii. Model Weight
5. In terms of Primary Keys, the following have been determined:
    a. Employee & Technician and Traffic Controller derivatives – Identification Number

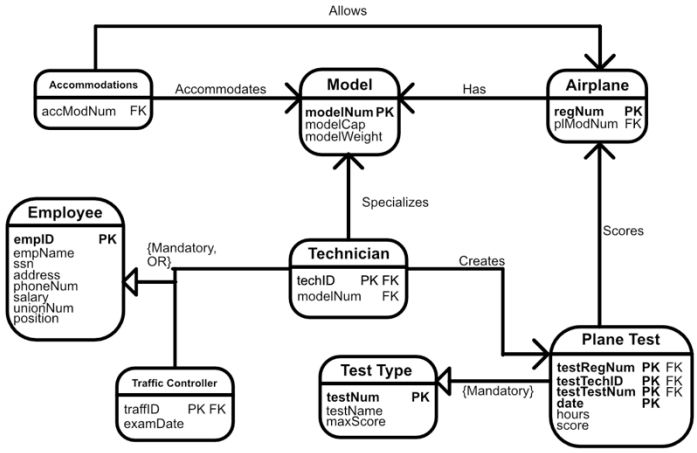
        i. All other attributes would not conform to the Entity Integrity Constraint, as all others but the Social Security Number can be shared amongst multiple employees
    b. Test Type – Test Number
        i. Assuming that the Test Number refers to a form of identification given to a given test type, rather than every instance of a test regardless of type, it serves as sufficient identification since the number wouldn't be repeated, yet two tests could share the same name
    c. Plane Test – Test Number, Plane Registration Number, Technician ID, Date
        i. All of these are necessary for the Entity Integrity Constraint to be satisfied, as the following conditions are true and the absence of one could lead to subsets that are identical:
            1. Multiple planes could have the same test performed on them, so the test number could be repeated
            2. A single plane could have multiple different tests, so the registration number could be repeated
            3. A technician could perform multiple tests, so the ID could be repeated
            4. Multiple identical tests could be run on the same plane and maybe even by the same technician at the same time, therefore the date could also be repeated
    d. Airport Accommodations – No Primary Key Required
        i. It would just serve as a list to reference and satisfy the Referential Integrity Constraint, making sure that all planes being handled are actually allowed on the airport.
    e. Airplane – Registration Number
        i. The registration number is a unique number that could not be shared, making it sufficient to differentiate each individual instance
    f. Model – Model Number
        i. This value varies according to the other two attributes, meaning that if either of them is different, this would be different. Therefore there is no need to add any more attributes to the primary key.
6. This is the resulting first iteration of the Data Model, focusing solely on its conceptual aspect:



a. The Entities Employee and Test Type become super classes that contain certain attributes to be used by their respective subclasses, Technician, Traffic Controller, and Plane Test.

Development towards the Global Logical Data Model

The ideal follow up to the Conceptual Data Model would be to begin designating attributes in the graphical representation, to identify how each entity relates to each other in terms of Relational Integrity and dependency, and at the same time it serves as a way to avoid data redundancy.



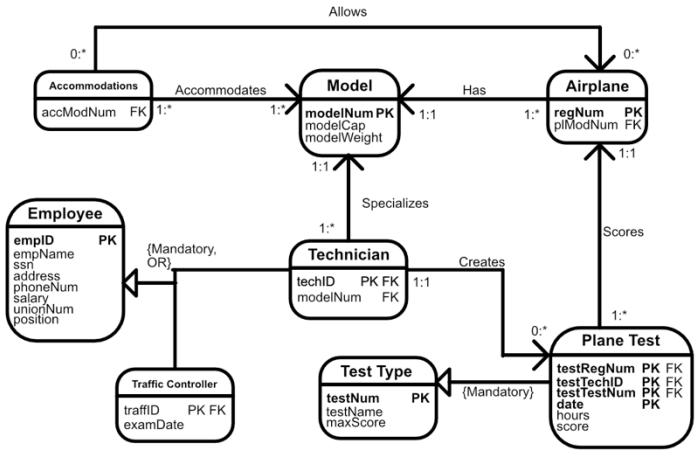
1. Within this new graphical representation, all attributes have been added to what are to be considered the optimal entities.
    a. Some tables do not require all information, so it would be best to exclude this information and store them separately.
        i. For example, the Plane Test entity requires the Technician's ID, but it doesn't ask any of their other attributes. Therefore this information has to be stored

within another table that can be referenced yet not have this information visible.

b. Some information has been distributed in order to make changes to information easier.
    i. For example, the Airplane entity requires a registration number and a model number, but each model number must be linked to a capacity and weight. A change to the capacity and/or weight becomes easier in a Model entity, where only one instance of each model is necessary.

2. Both Primary Keys and Foreign Keys have been identified.
    a. Foreign keys chosen are to avoid redundancy and/or errors.
        i. These foreign keys are able to assure that the airplane models referenced exist and are allowed on the airport, that the particular airplane, technician, and test instances exists by checking the registration number, ID, and test number.
    b. Currently, potential constraints are defined as follows:
        i. Given the information, an employee must have a position of Technician of Traffic Controller
        ii. Each Plane Test instance can only be of one Test Type and be scored by one Technician, both of which must exist, and be scored
        iii. Each Airplane instance must only be of one Model, which must exist and be accommodated for by the airport
        iv. Each Model can only have one Capacity and one Weight
        v. Each Test Type must have a name
        vi. Every employee must have a different SSN and ID
        vii. All other non-primary key attributes can be shared amongst various instances of entities.
        viii. No attributes seem to need the option to contain a NULL value
        ix. Some attributes may have default values:
            1. MaxScore can default to 100
            2. Salary can default to a starting salary for new employees

With some of these things in mind, the third instance of the graphical representation is as follows:



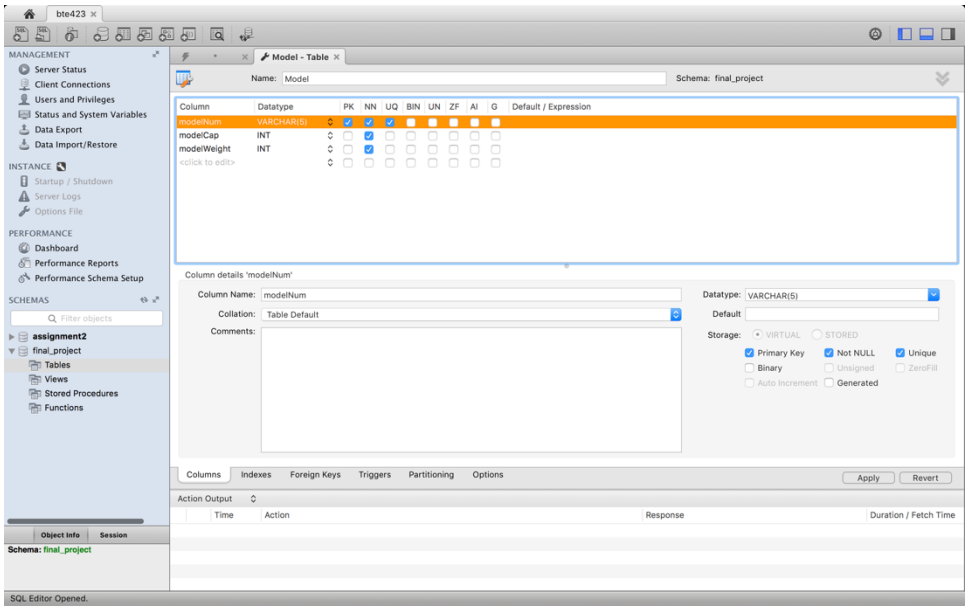
With this model in place, the following statements are true:

1. The Accommodations list accommodates one to many Models
2. Each Model is accommodated by one or more Accommodations lists (in this case study, there is only one list)
3. Each Airplane has only one model
4. Each model can belong to one to many Airplanes
5. Each Technician specializes in one and only one Model
6. A Model can be the specialization of one to many Technicians
7. Every Technician can create zero to many Plane Tests
8. Every Plane Test must be created by one and only one Technician
9. Each Plane Test scores one and only one Airplane
10. Each Airplane is scored by one to many different Plane Tests
11. Each employee must be specialized as a Technician or Traffic Controller (assuming the context of this case study)
12. Each Plane Test must come from a Test Type

MySQL Integration

While implementing this model into a MYSQL database, it was ideal to begin with by creating the Model, Employee, and Test Type tables first, and then branching off from their to ensure referential integrity and foreign keys to be obtained from these sources.

1. Model:
    a. Screenshot:



    b. Query:

```
CREATE TABLE `Model` (

 `modelNum` varchar(5) NOT NULL,

 `modelCap` int(11) NOT NULL,
```

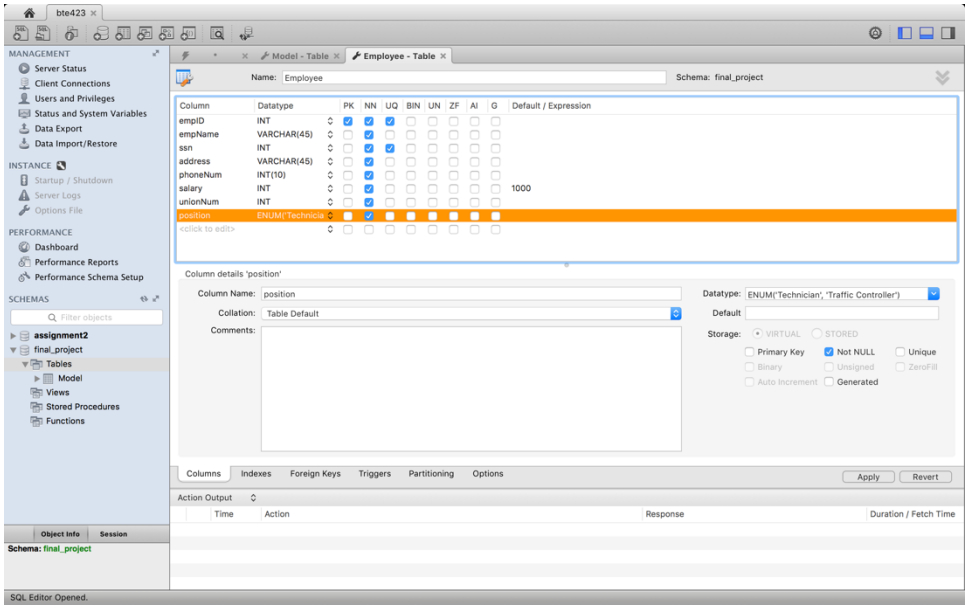
`modelWeight` int(11) NOT NULL,

PRIMARY KEY (`modelNum`),

UNIQUE KEY `modelNum_UNIQUE` (`modelNum`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

2. Employee:
    a. Screenshot:



    b. Query:

CREATE TABLE `final_project`.`Employee` (

`empID` INT NOT NULL,

`empName` VARCHAR(45) NOT NULL,

`ssn` INT NOT NULL,

`address` VARCHAR(45) NOT NULL,

`phoneNum` VARCHAR(10) NOT NULL,

`salary` INT NULL DEFAULT 900,

`unionNum` INT NOT NULL,

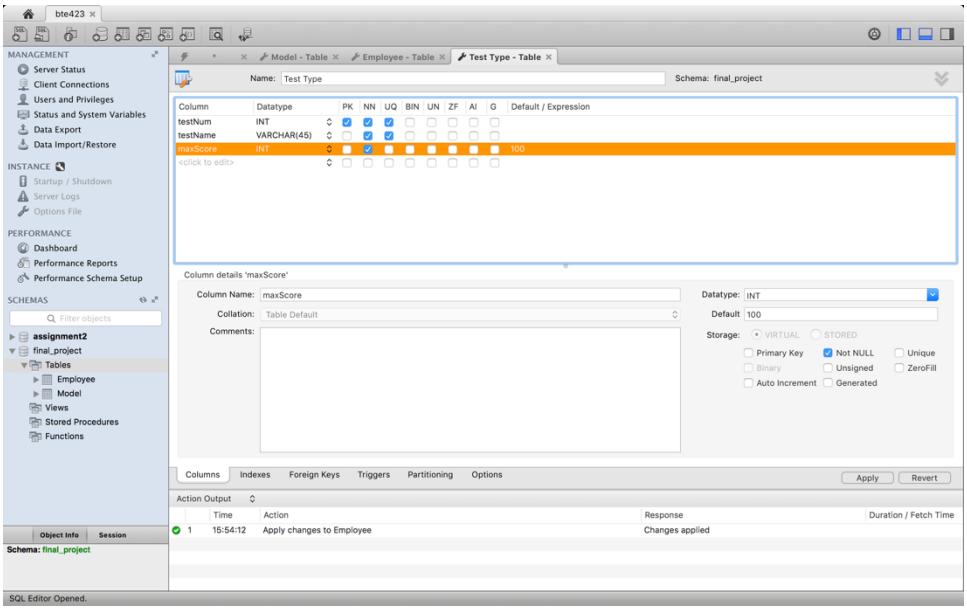
`position` ENUM('Technician', 'Traffic Controller') NOT NULL,

PRIMARY KEY (`empID`),

UNIQUE INDEX `empID_UNIQUE` (`empID` ASC),

UNIQUE INDEX `ssn_UNIQUE` (`ssn` ASC));

3. Test Type:
    a. Screenshot:

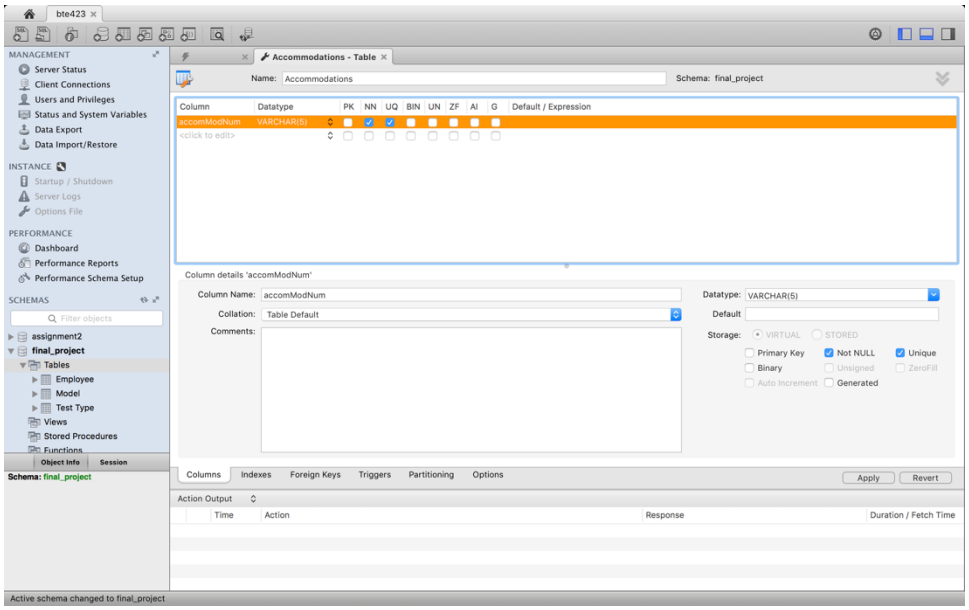


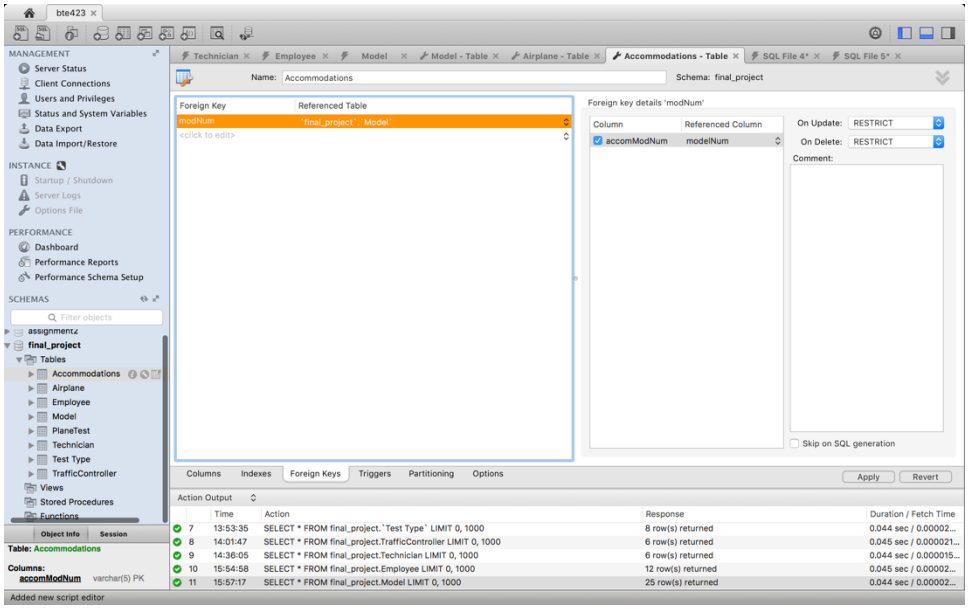
b. Query:

```
CREATE TABLE `final_project`.`Test Type` (
  `testNum` INT NOT NULL,
  `testName` VARCHAR(45) NOT NULL,
  `maxScore` INT NOT NULL DEFAULT 100,
  PRIMARY KEY (`testNum`),
  UNIQUE INDEX `testNum_UNIQUE` (`testNum` ASC),
  UNIQUE INDEX `testName_UNIQUE` (`testName` ASC));
```

With these set, the following tables can be assembled:

1. Accommodations
    a. Screenshots:

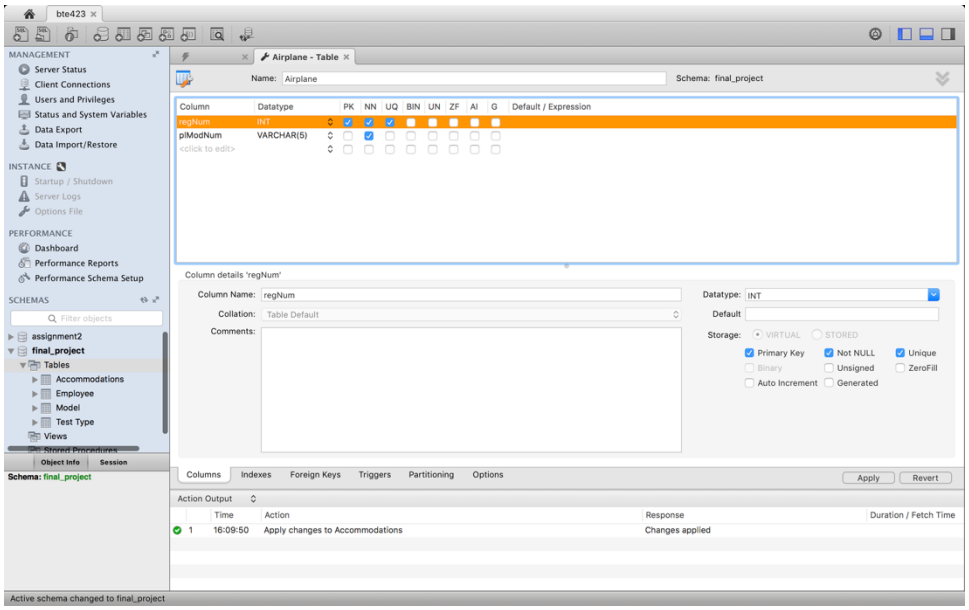


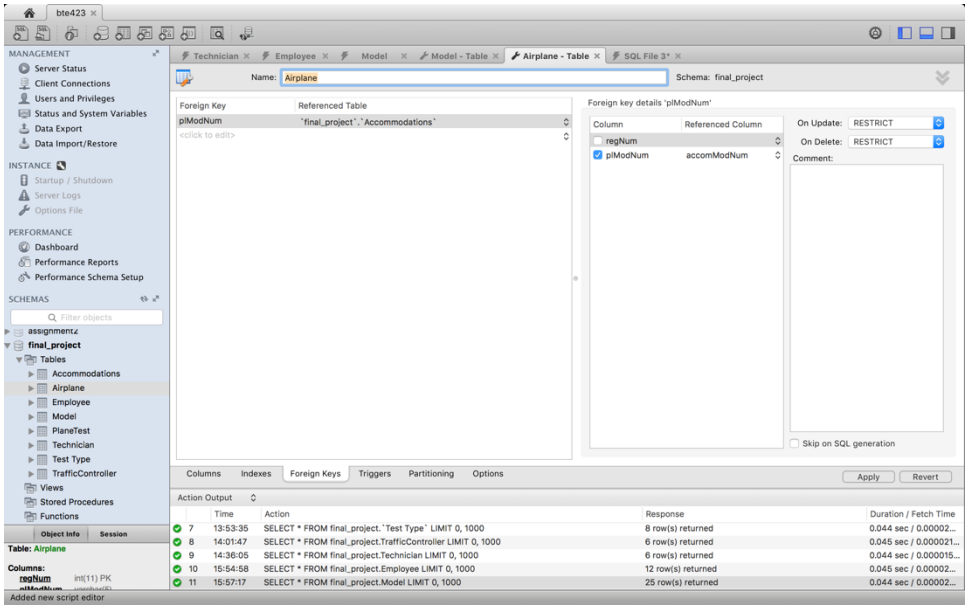


b. Query:

```
CREATE TABLE `Accommodations` (
 `accomModNum` varchar(5) NOT NULL,
 UNIQUE KEY `idnew_table_UNIQUE` (`accomModNum`),
 CONSTRAINT `modNum` FOREIGN KEY (`accomModNum`) REFERENCES `Model` (`modelNum`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

2. Airplane:
   a. Screenshots:

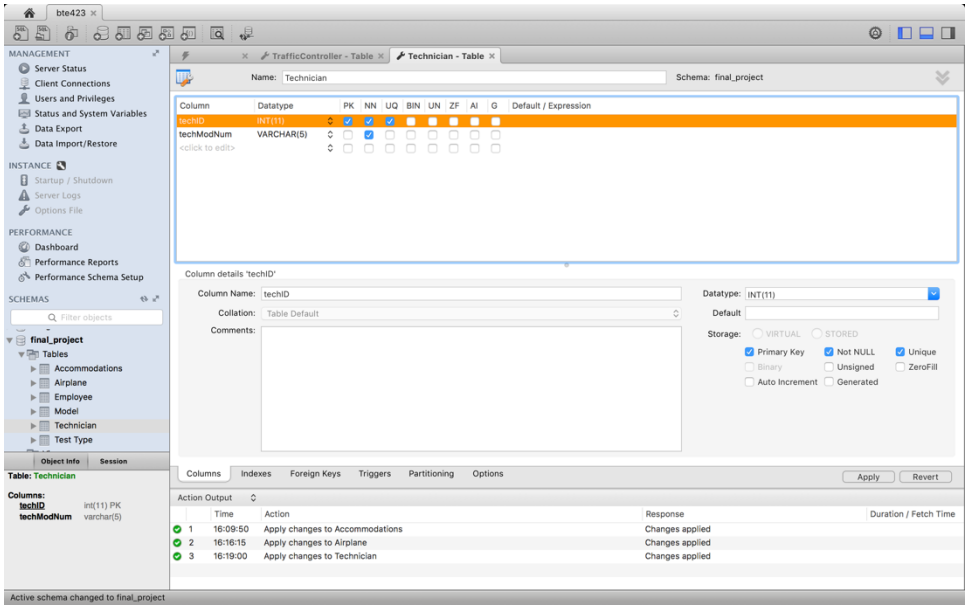


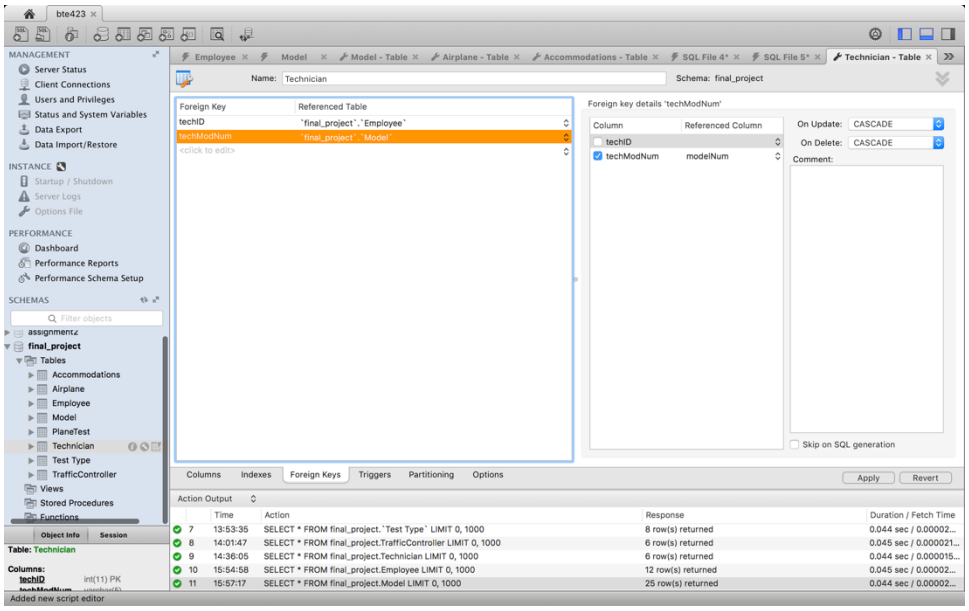


b. Query:

```
CREATE TABLE `Airplane` (
 `regNum` int(11) NOT NULL,
 `plModNum` varchar(5) NOT NULL,
 PRIMARY KEY (`regNum`),
 UNIQUE KEY `regNum_UNIQUE` (`regNum`),
 KEY `plModNum_idx` (`plModNum`),
 CONSTRAINT `plModNum` FOREIGN KEY (`plModNum`) REFERENCES `Accommodations` (`accomModNum`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

3. Technician:
    a. Screenshots:





    b. Query:

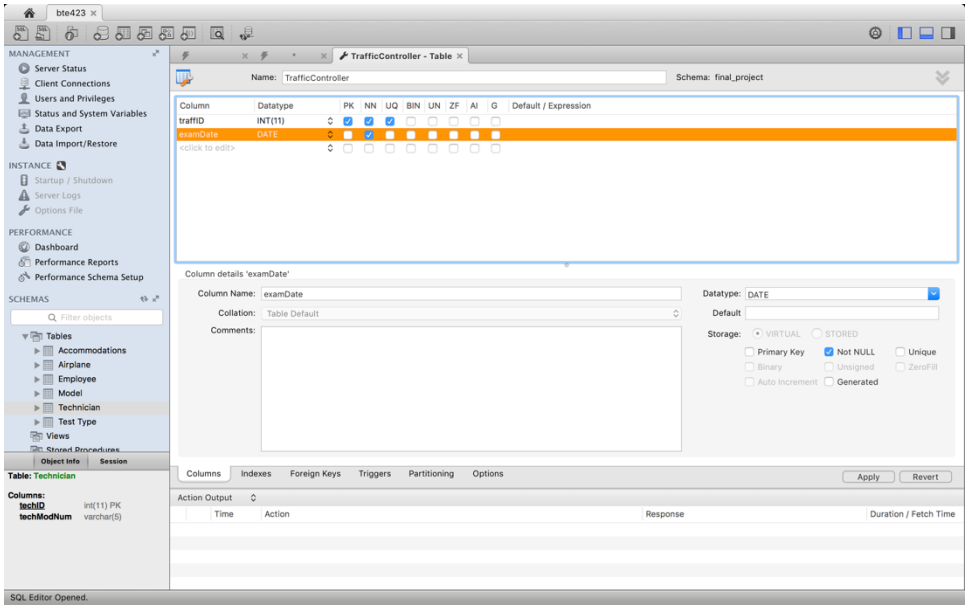
```
CREATE TABLE `Technician` (
  `techID` int(11) NOT NULL,
  `techModNum` varchar(5) NOT NULL,
  PRIMARY KEY (`techID`),
  UNIQUE KEY `techID_UNIQUE` (`techID`),
  KEY `techModNum_idx` (`techModNum`),
```

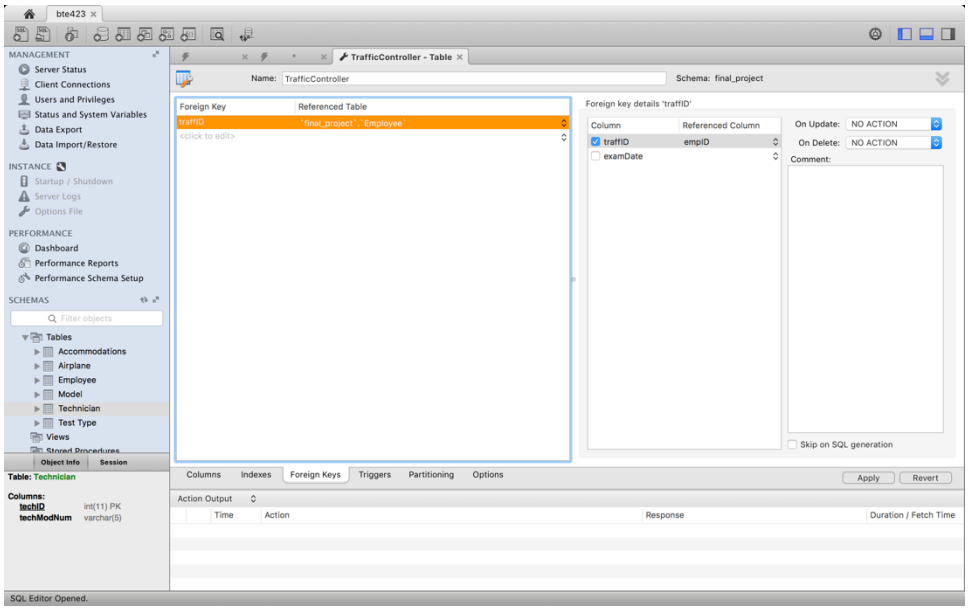
CONSTRAINT `techID` FOREIGN KEY (`techID`) REFERENCES `Employee` (`empID`) ON DELETE NO ACTION ON UPDATE NO ACTION,

CONSTRAINT `techModNum` FOREIGN KEY (`techModNum`) REFERENCES `Model` (`modelNum`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

4. Traffic Controller
    a. Screenshots:





    b. Query:

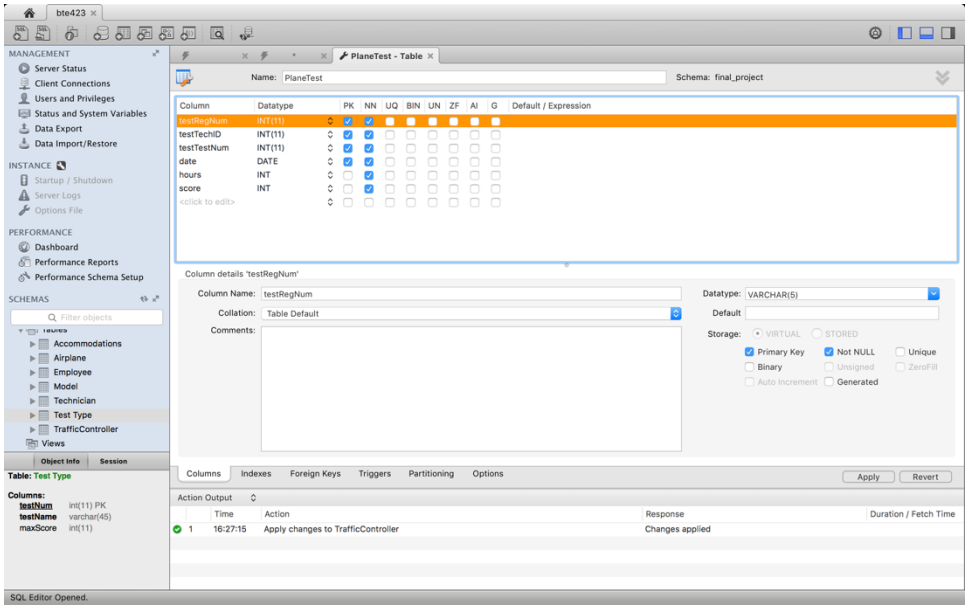
CREATE TABLE `final_project`.`TrafficController` (

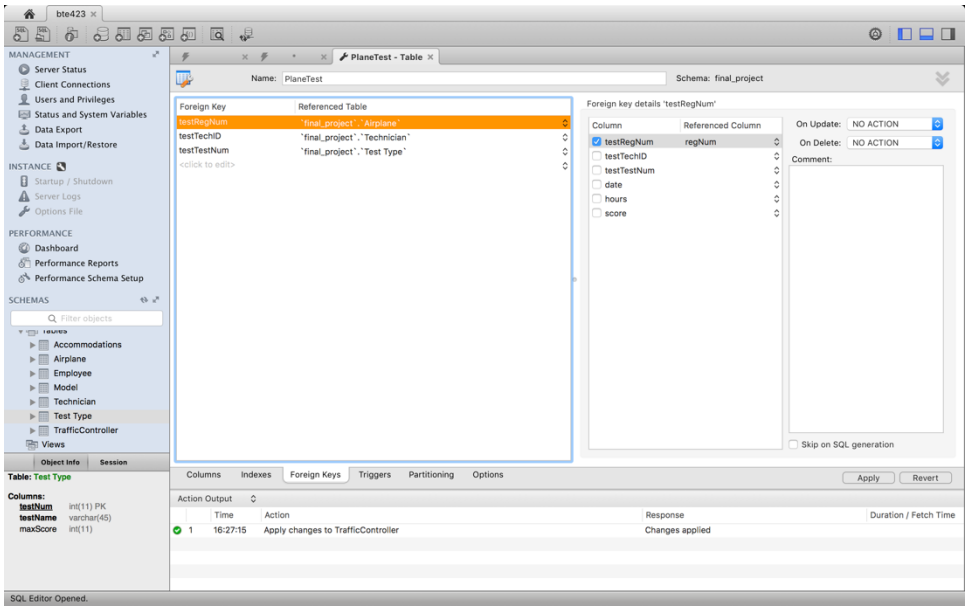
`traffID` INT(11) NOT NULL,

```
`examDate` DATE NOT NULL,
PRIMARY KEY (`traffID`),
UNIQUE INDEX `traffID_UNIQUE` (`traffID` ASC),
CONSTRAINT `traffID`
 FOREIGN KEY (`traffID`)
 REFERENCES `final_project`.`Employee` (`empID`)
 ON DELETE NO ACTION
 ON UPDATE NO ACTION);
```

Now the final table can be created.

1. Plane Test
    a. Screenshots:





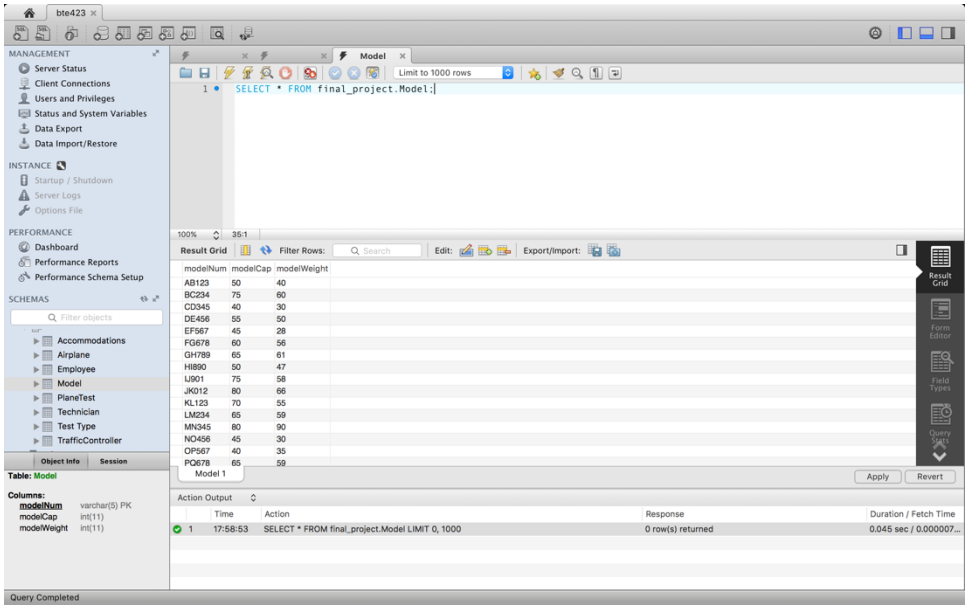
b. Query:

```
CREATE TABLE `final_project`.`PlaneTest` (
  `testRegNum` INT(11) NOT NULL,
  `testTechID` INT(11) NOT NULL,
  `testTestNum` INT(11) NOT NULL,
  `date` DATE NOT NULL,
  `hours` INT NOT NULL,
  `score` INT NOT NULL CHECK (score BETWEEN 0 AND 100),
  PRIMARY KEY (`testRegNum`, `testTechID`, `testTestNum`, `date`),
  INDEX `testTechID_idx` (`testTechID` ASC),
  INDEX `testTestNum_idx` (`testTestNum` ASC),
  CONSTRAINT `testRegNum`
    FOREIGN KEY (`testRegNum`)
    REFERENCES `final_project`.`Airplane` (`regNum`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `testTechID`
    FOREIGN KEY (`testTechID`)
    REFERENCES `final_project`.`Technician` (`techID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `testTestNum`
```

```
FOREIGN KEY (`testTestNum`)
REFERENCES `final_project`.`Test Type` (`testNum`)
ON DELETE NO ACTION
ON UPDATE NO ACTION);
```

With all these tables in place, sample data can be added to them in order to test the sample queries assigned to this case. Information will be added in the same order as the tables were created in order to allow appropriate referencing through foreign keys.

1. Model
    a. 25 initial models were added to the table, ranging in various capacities and weights, and each with their own unique Model Number.
    b. Screenshot:

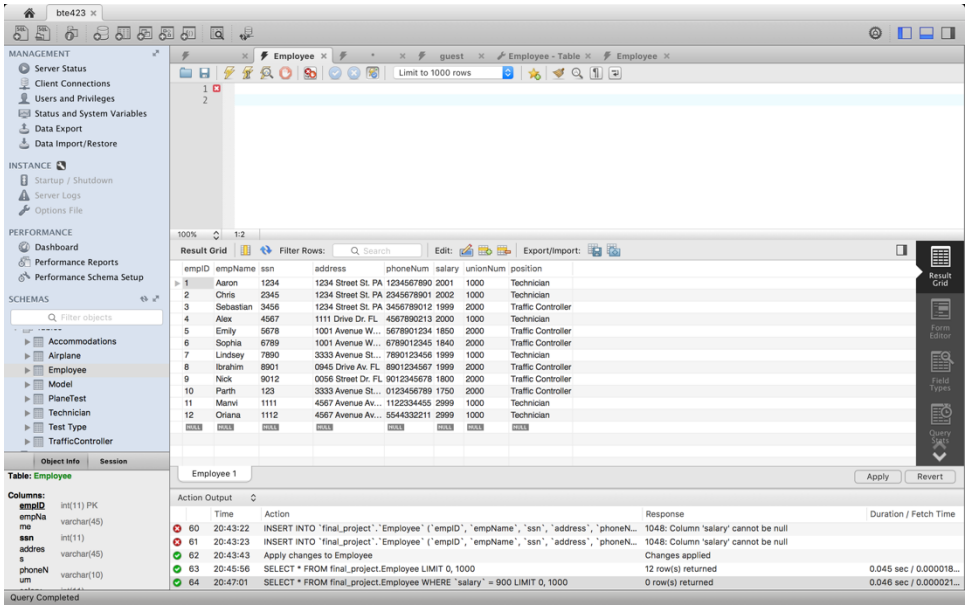


    c. Query:

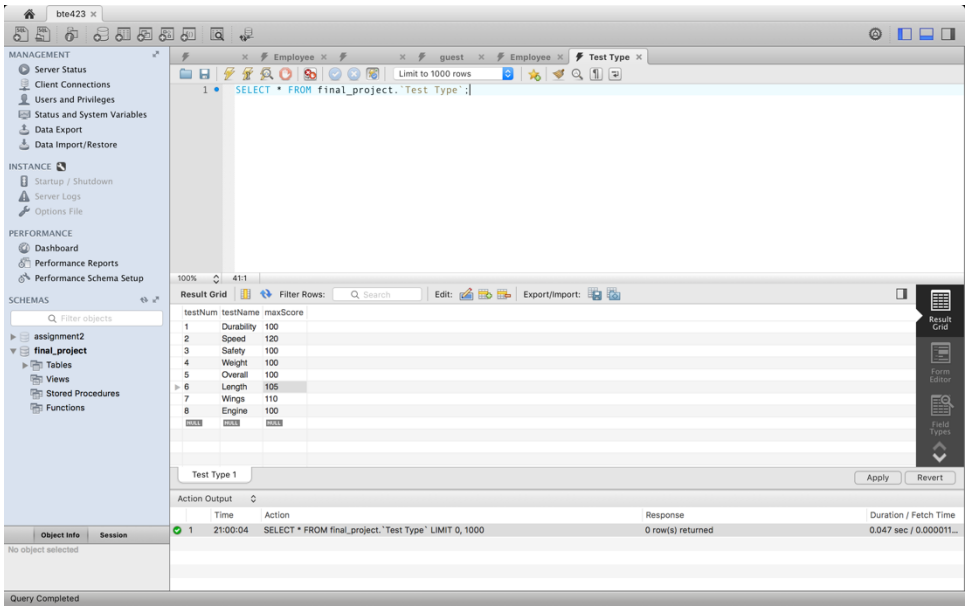
```
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('AB123', '50', '40');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('BC234', '75', '60');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('CD345', '40', '30');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('DE456', '55', '50');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('EF567', '45', '28');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('FG678', '60', '56');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('GH789', '65', '61');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('HI890', '50', '47');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('IJ901', '75', '58');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('JK012', '80', '66');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('KL123', '70', '55');
```

```
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('LM234', '65', '59');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('MN345', '80', '90');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('NO456', '45', '30');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('OP567', '40', '35');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('PQ678', '65', '59');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('QR789', '50', '56');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('RS890', '50', '52');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('ST901', '40', '40');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('TU012', '85', '76');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('UV123', '70', '90');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('VW234', '60', '100');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('WX345', '45', '34');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('XY456', '50', '46');
INSERT INTO `final_project`.`Model` (`modelNum`, `modelCap`, `modelWeight`) VALUES ('YZ567', '55', '42');
```

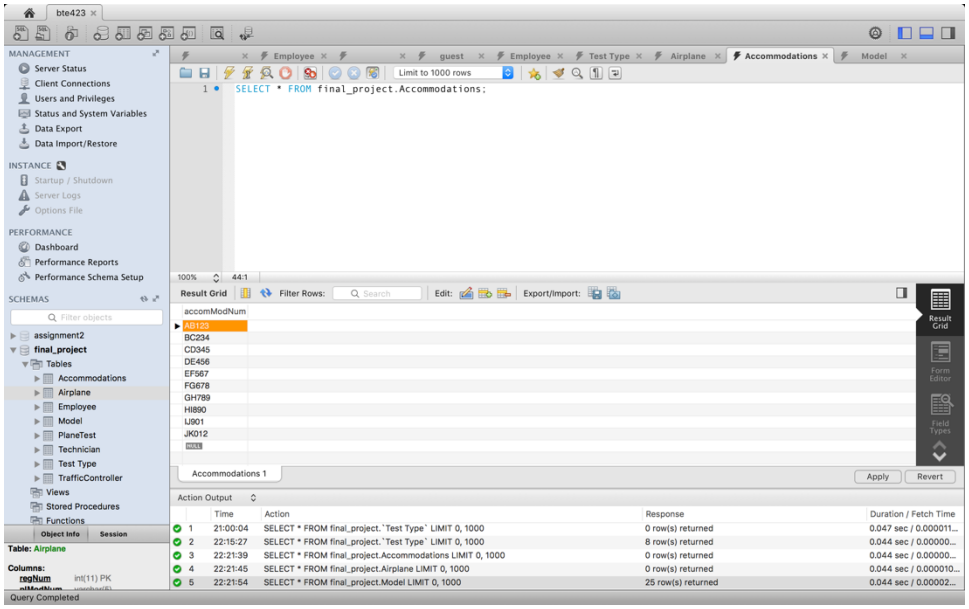
2. Employee
    a. 12 initial employees were generated, half being Traffic Controllers and half being Technicians.
    b. Screenshot:



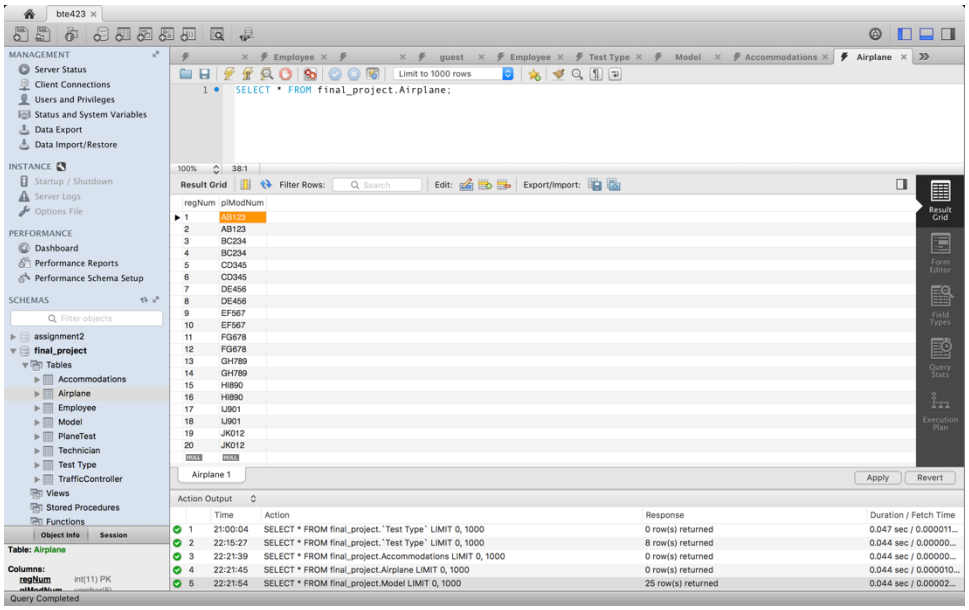
3. Test Type
    a. 8 initial test types were generated.



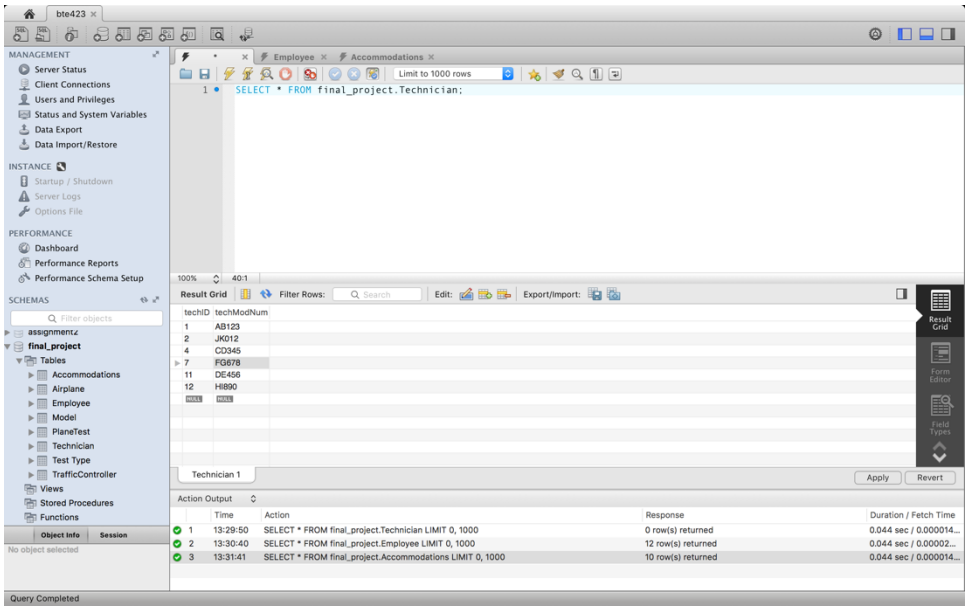
4. Accommodations
    a. 10 of the models have been added to the accommodations list, filtering out the models that can be used by the Airplane table later on.



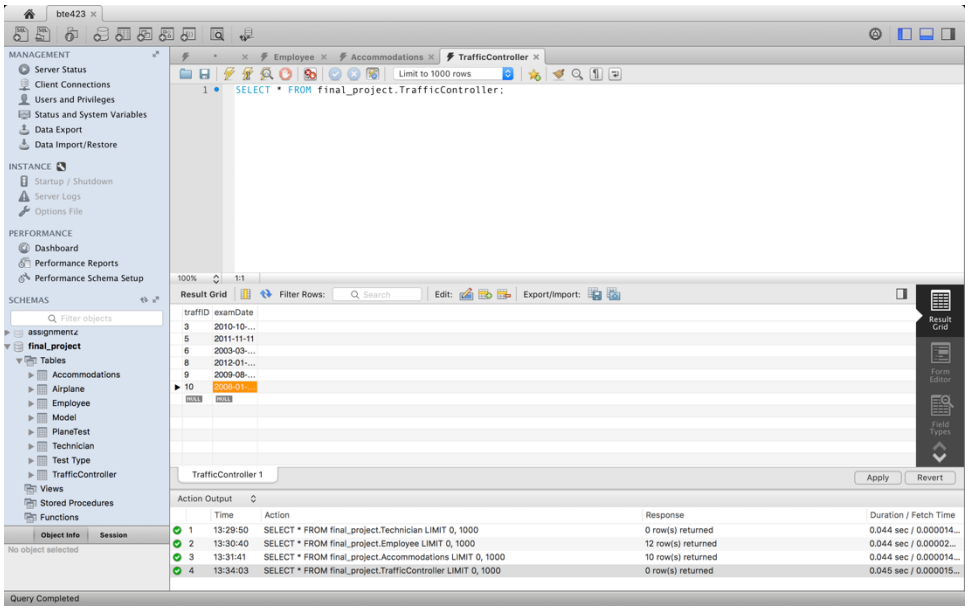
5. Airplane
    a. 2 airplanes were created for each model accommodated by the airport



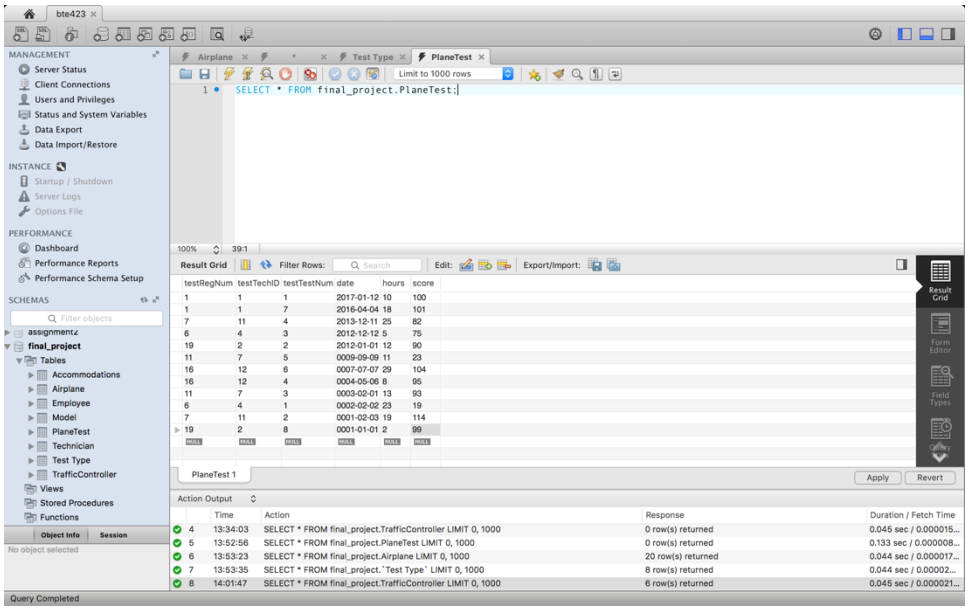
6. Technician
    a. All 6 employees with the Technician position were assigned a Model of expertise.



7. Traffic Controller
    a. All 6 employees with the Traffic Controller position were assigned an exam date.



8. Plane Test
    a. Two tests were logged for each plane recorded at the airport.



Sample Queries

1. Two queries must be executed in order to create a Technician, one to create the employee and another to assign their expertise.
    a. INSERT INTO `final_project`.`Employee` (`empID`, `empName`, `ssn`, `address`, `phoneNum`, `salary`, `unionNum`, `position`) VALUES ('13', 'New', '9876', '3333 Avenue St. MD', '9999999999', '1111', '1000', 'Technician');

    b. INSERT INTO `final_project`.`Technician` (`techID`, `techModNum`) VALUES ('13', 'AB123');

2. DELETE FROM `final_project`.`Airplane` WHERE `regNum`='20';

3. UPDATE `final_project`.`Technician` SET `techModNum`='BC234' WHERE `techID`='13';

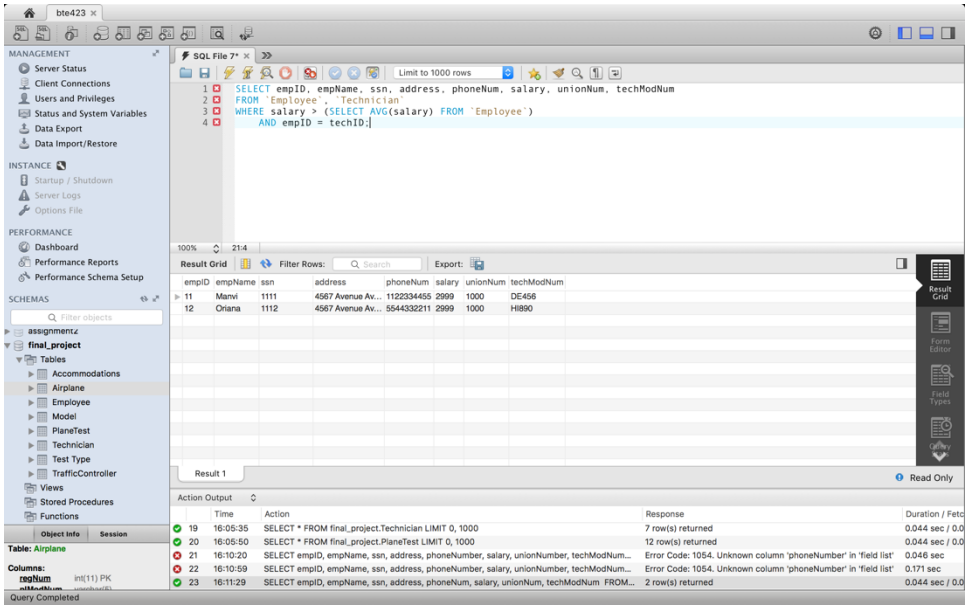
4. To make sure that the syntax of the following query works, a second query calculating the average salary is prepared in b.

   a. SELECT empID, empName, ssn, address, phoneNum, salary, unionNum, techModNum

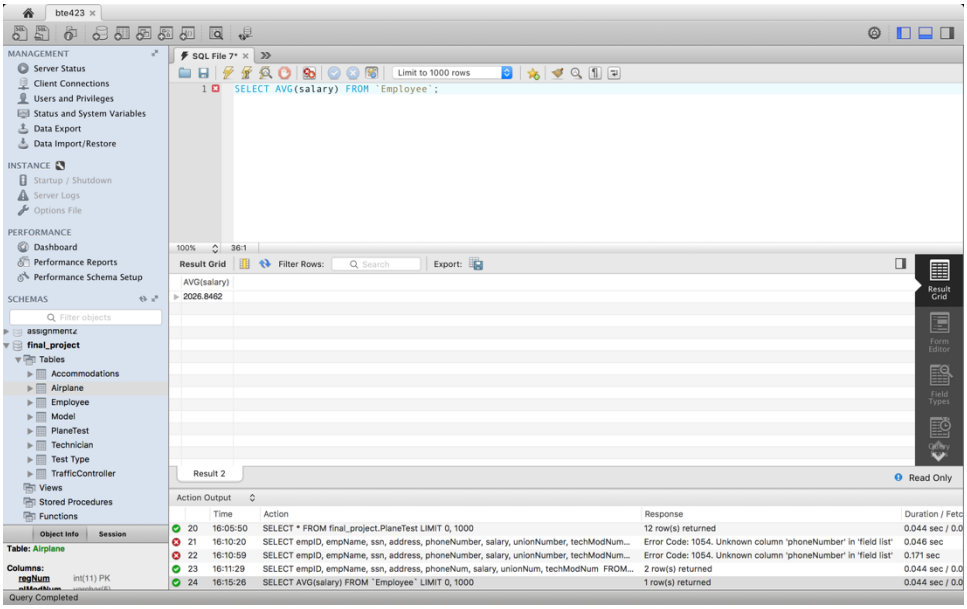
      FROM `Employee`, `Technician`

      WHERE salary > (SELECT AVG(salary) FROM `Employee`)

      AND empID = techID;



   b. SELECT AVG(salary) FROM `Employee`;

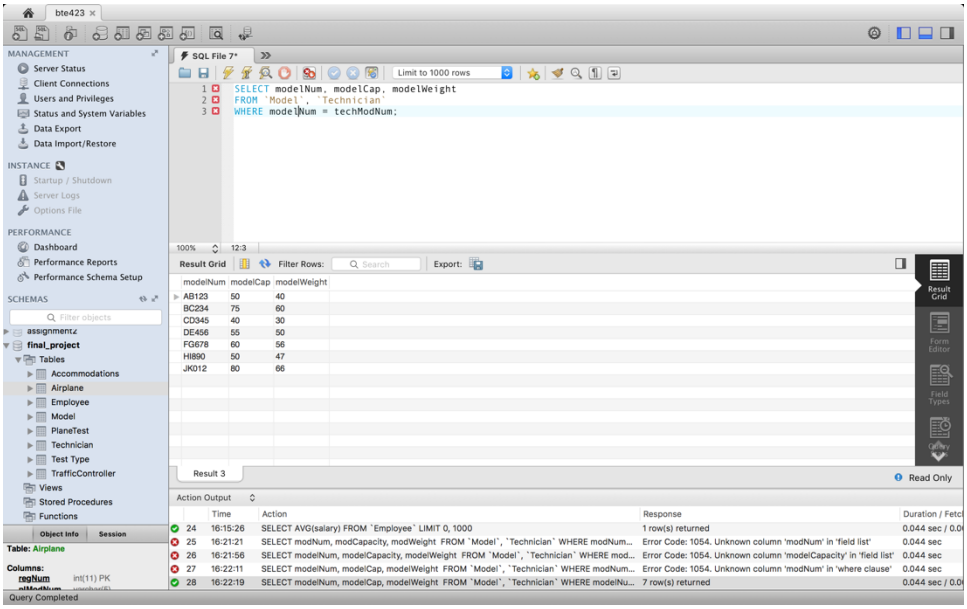


The returned value is the correct average, and would result in providing the returned values in a.

5. SELECT modelNum, modelCap, modelWeight

   FROM `Model`, `Technician`

   WHERE modelNum = techModNum;

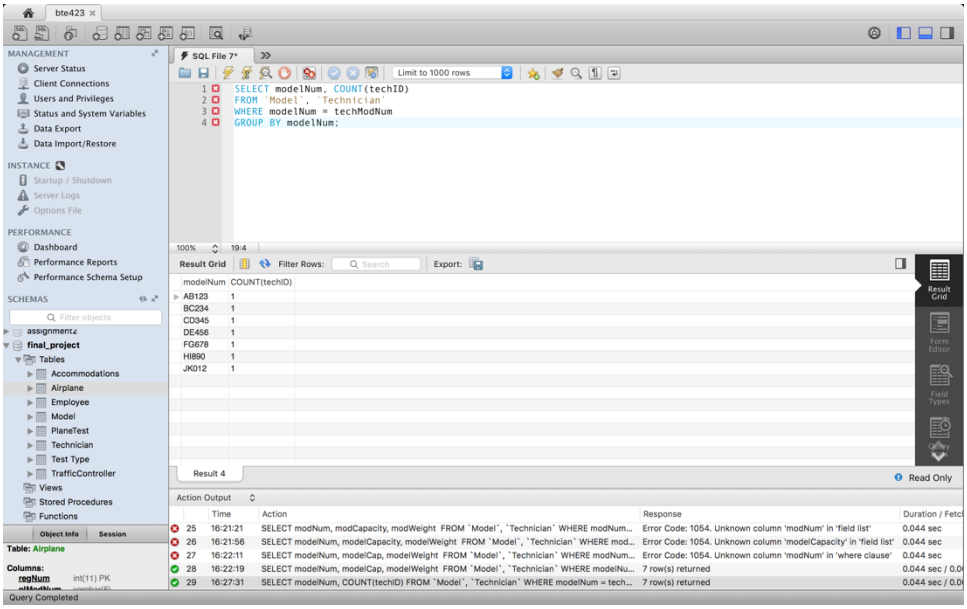


6. SELECT modelNum, COUNT(techID)

   FROM `Model`, `Technician`

   WHERE modelNum = techModNum

   GROUP BY modelNum;



7. SELECT testRegNum, testNum, testTechID, date, hours, score, testName, maxScore

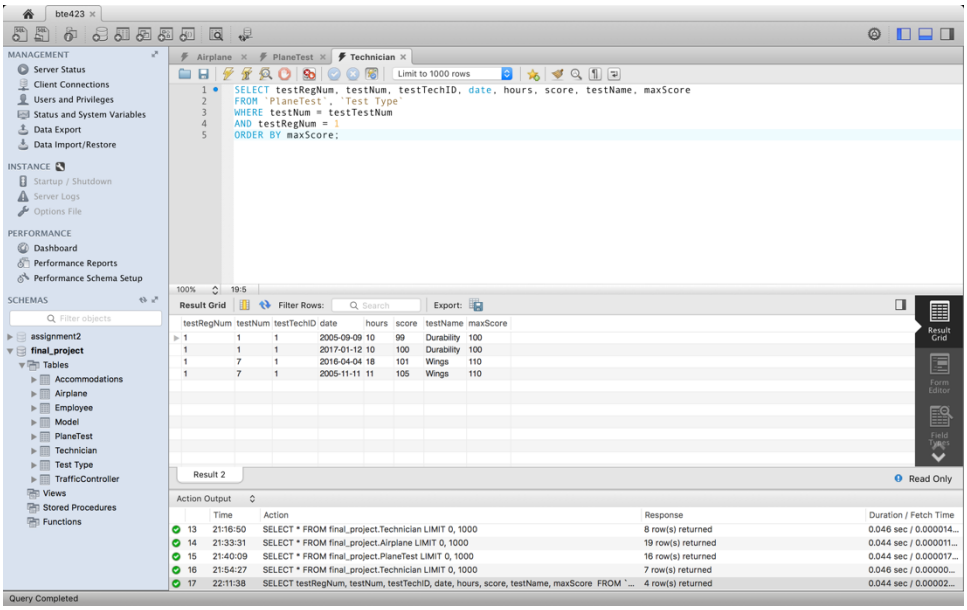
   FROM `PlaneTest`, `Test Type`

WHERE testNum = testTestNum

AND testRegNum = 1

ORDER BY maxScore;

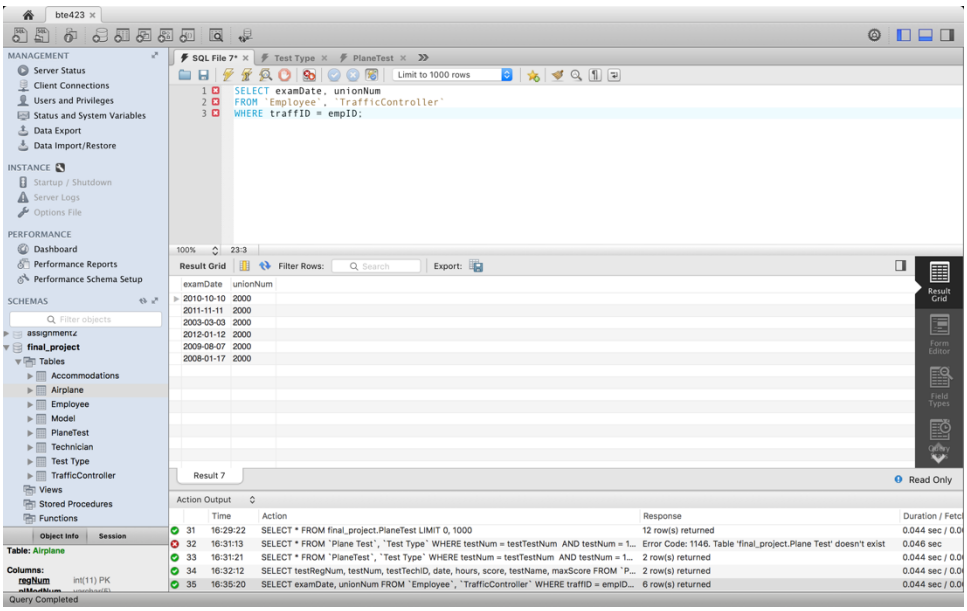
The screenshot is from a retest, further down the line, therefore more results are present.



8. SELECT examDate, unionNum

FROM `Employee`, `TrafficController`

WHERE traffID = empID;

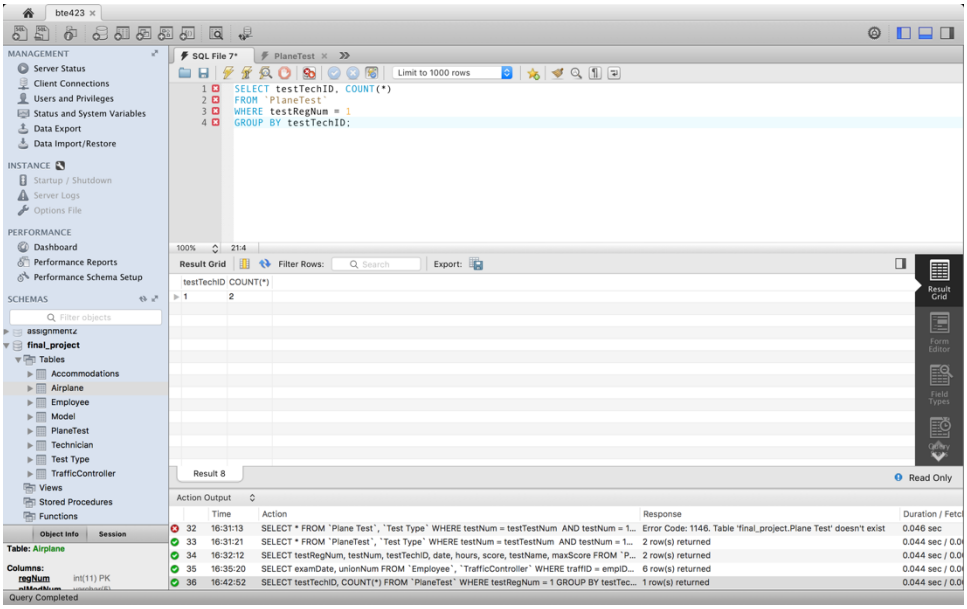


9. SELECT testTechID, COUNT(*)

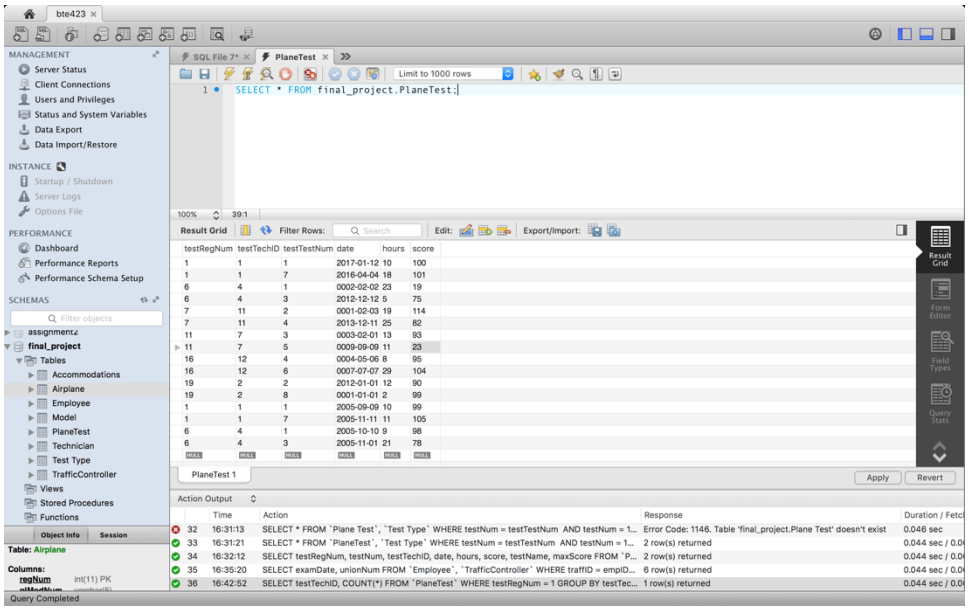
FROM `PlaneTest`

WHERE testRegNum = 1

GROUP BY testTechID;



10. For this query to show results, it was necessary to add more instances to the table. A total of four more tests were added, all of which have dates that are included in the time frame being asked for.
    a.



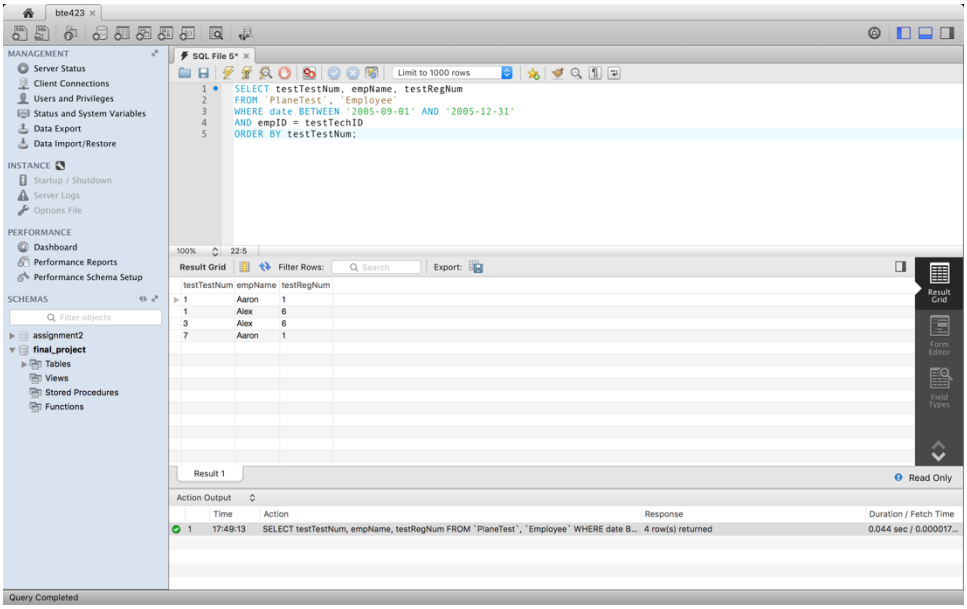
    b. SELECT testTestNum, empName, testRegNum

       FROM `PlaneTest`, `Employee`

       WHERE date BETWEEN '2005-09-01' AND '2005-12-31'

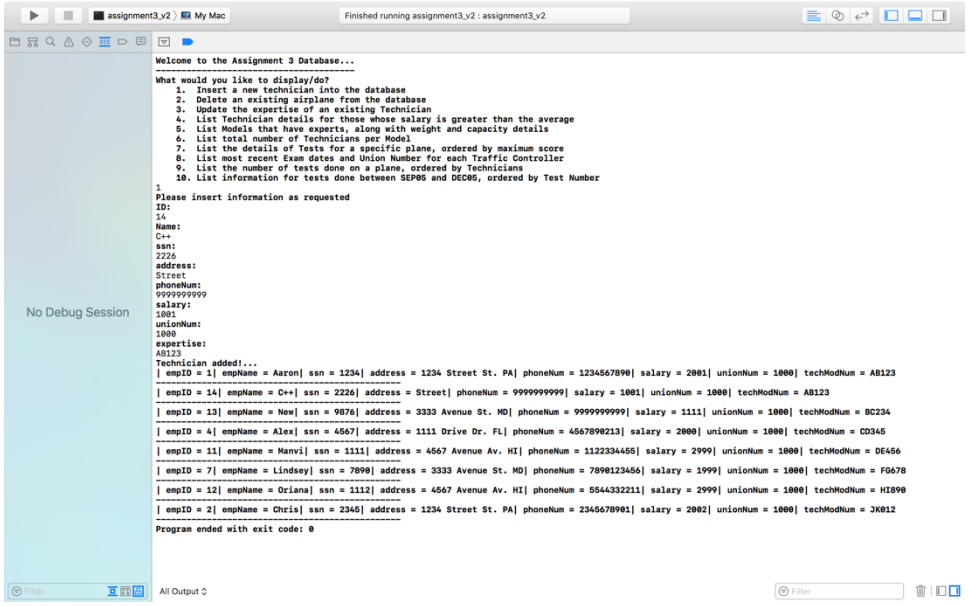
AND empID = testTechID

ORDER BY testTestNum;

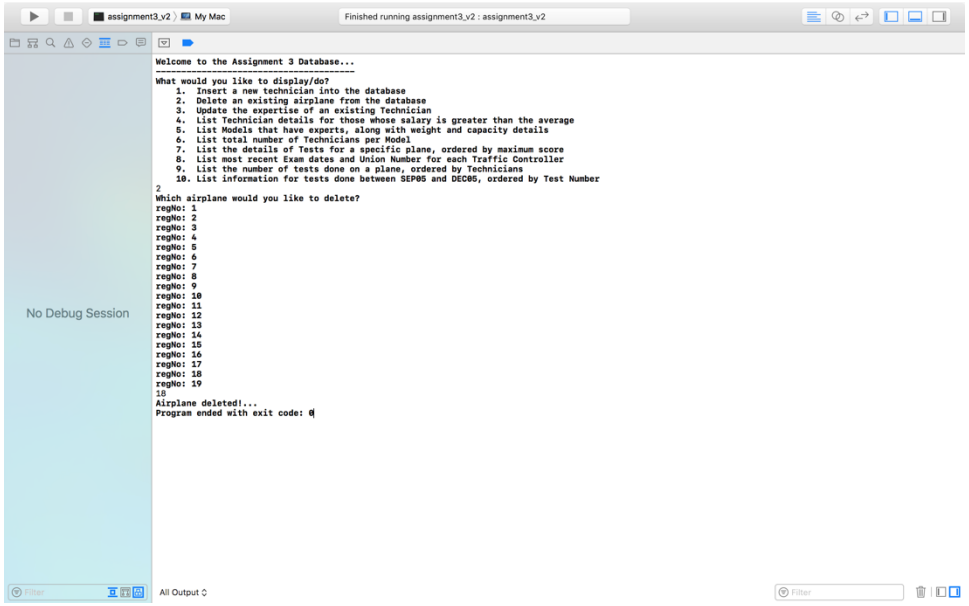


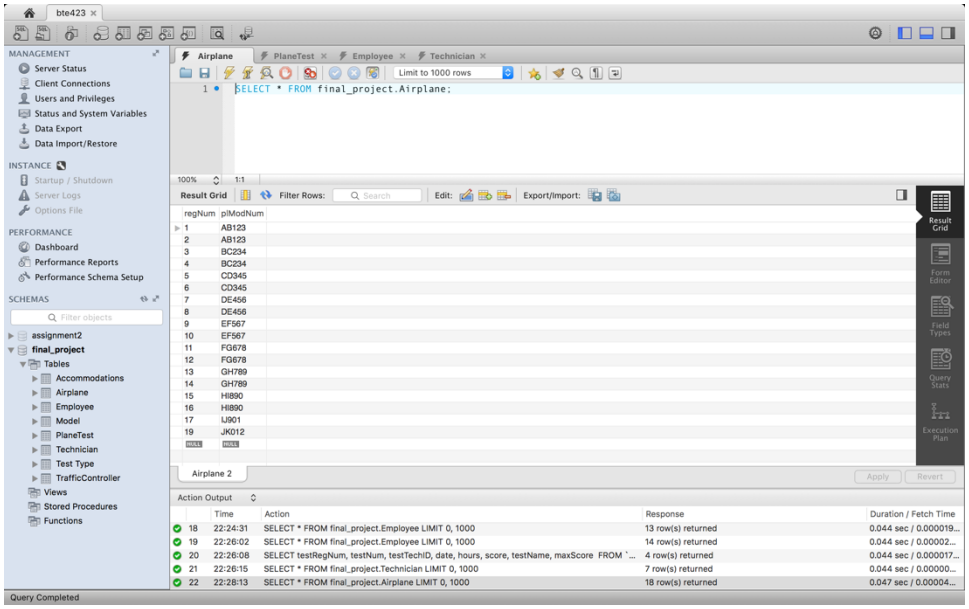
Console Sample Queries:

1.

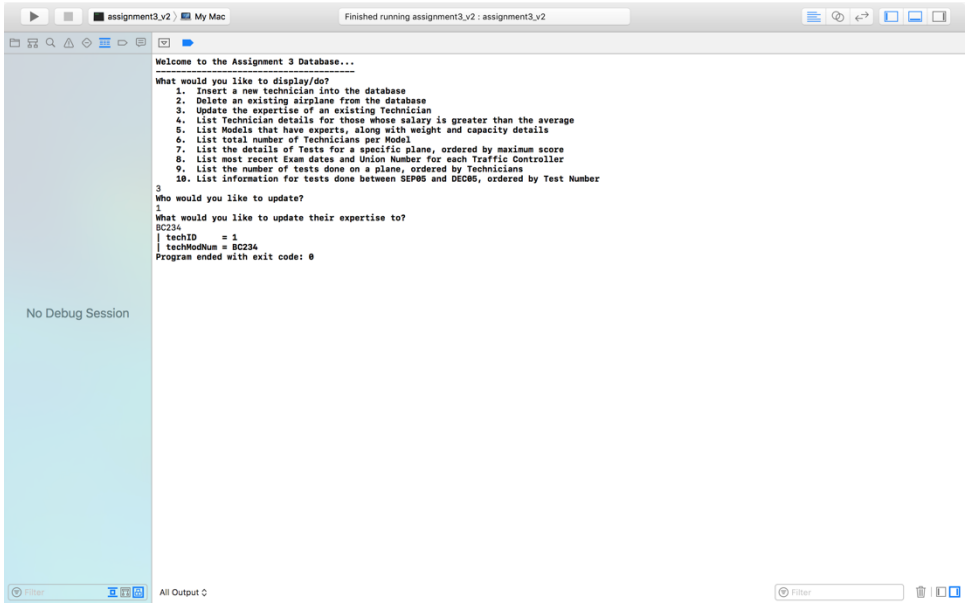


2.

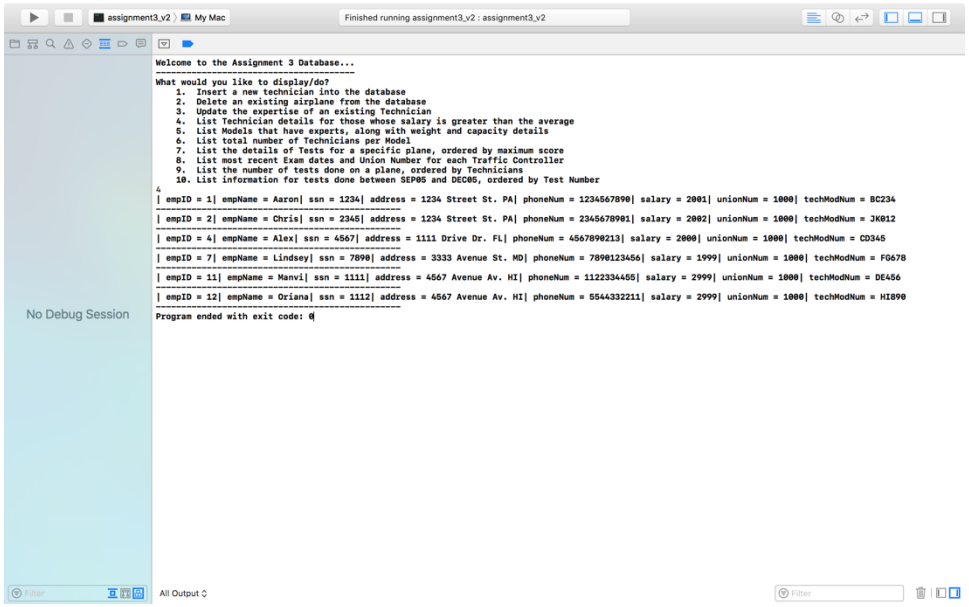

assignment3_v2 〉 My Mac
Finished running assignment3_v2 : assignment3_v2
No Debug Session
All Output
Welcome to the Assignment 3 Database...
What would you like to display/do?
1. Insert a new technician into the database
2. Delete an existing airplane from the database
3. Update the expertise of an existing Technician
4. List Technician details for those whose salary is greater than the average
5. List Models that have experts, along with weight and capacity details
6. List total number of Technicians per Model
7. List the details of Tests for a specific plane, ordered by maximum score
8. List most recent Exam dates and Union Number for each Traffic Controller
9. List the number of tests done on a plane, ordered by Technicians
10. List information for tests done between SEP05 and DEC05, ordered by Test Number
3
Which airplane would you like to delete?
regNo: 1
regNo: 2
regNo: 3
regNo: 4
regNo: 5
regNo: 6
regNo: 7
regNo: 8
regNo: 9
regNo: 10
regNo: 11
regNo: 12
regNo: 13
regNo: 14
regNo: 15
regNo: 16
regNo: 17
regNo: 18
regNo: 19
18
Airplane deleted!...
Program ended with exit code: 0

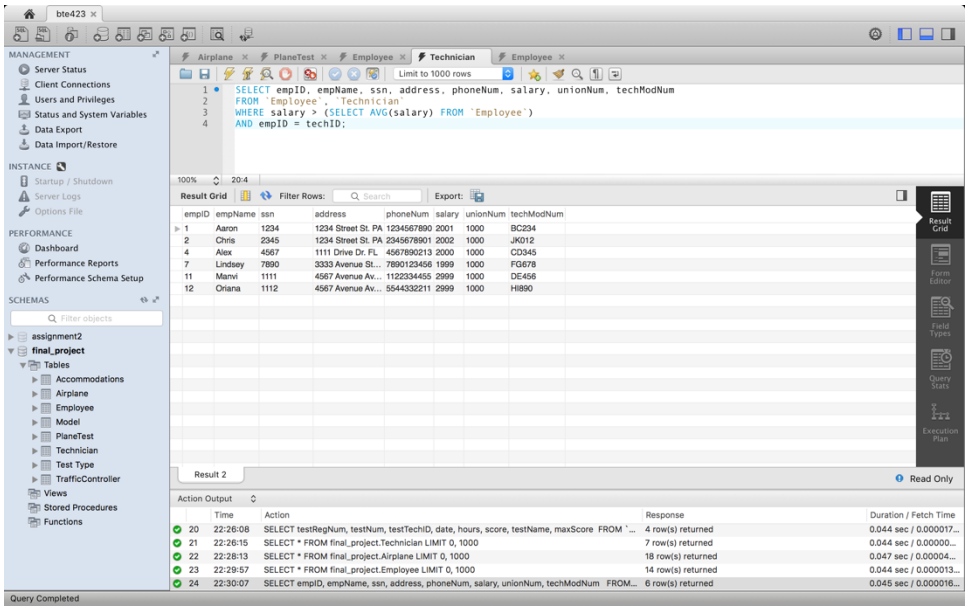

bte423
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
PERFORMANCE
Dashboard
Performance Reports
Performance Schema Setup
SCHEMAS
assignment2
final_project
Tables
Accommodations
Airplane
Employee
Model
PlaneTest
Technician
Test Type
TrafficController
Views
Stored Procedures
Functions
Airplane
PlaneTest
Employee
Technician
Limit to 1000 rows
SELECT * FROM final_project.Airplane;
Result Grid
Filter Rows:
Edit:
Export/Import:
regNum
plModNum
Airplane 2
Apply
Revert
Action Output
Time
Action
Response
Duration / Fetch Time
18 22:24:31 SELECT * FROM final_project.Employee LIMIT 0, 1000 13 row(s) returned 0.044 sec / 0.000019...
19 22:26:02 SELECT * FROM final_project.Employee LIMIT 0, 1000 14 row(s) returned 0.044 sec / 0.00002...
20 22:26:08 SELECT testRegNum, testNum, testTechID, date, hours, score, testName, maxScore FROM `... 4 row(s) returned 0.044 sec / 0.000017...
21 22:26:15 SELECT * FROM final_project.Technician LIMIT 0, 1000 7 row(s) returned 0.044 sec / 0.00000...
22 22:28:13 SELECT * FROM final_project.Airplane LIMIT 0, 1000 18 row(s) returned 0.047 sec / 0.00004...
Query Completed

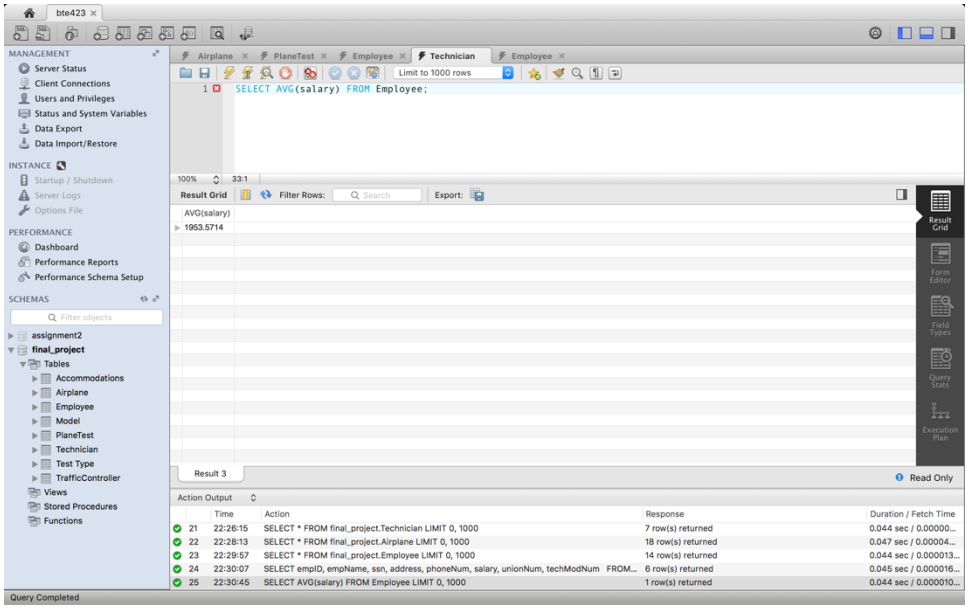
3.


assignment3_v2 〉 My Mac
Finished running assignment3_v2 : assignment3_v2
No Debug Session
All Output
Welcome to the Assignment 3 Database...
What would you like to display/do?
1. Insert a new technician into the database
2. Delete an existing airplane from the database
3. Update the expertise of an existing Technician
4. List Technician details for those whose salary is greater than the average
5. List Models that have experts, along with weight and capacity details
6. List total number of Technicians per Model
7. List the details of Tests for a specific plane, ordered by maximum score
8. List most recent Exam dates and Union Number for each Traffic Controller
9. List the number of tests done on a plane, ordered by Technicians
10. List information for tests done between SEP05 and DEC05, ordered by Test Number
3
Who would you like to update?
1
What would you like to update their expertise to?
BC234
| techID = 1
| techModNum = BC234
Program ended with exit code: 0

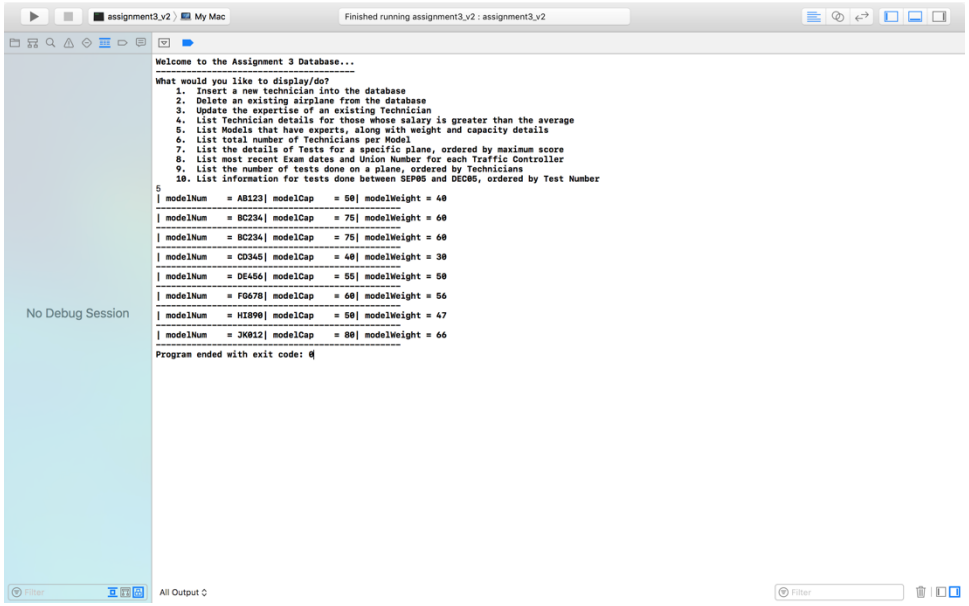
4.


assignment3_v2 〉 My Mac
Finished running assignment3_v2 : assignment3_v2
No Debug Session
All Output
Welcome to the Assignment 3 Database...
What would you like to display/do?
1. Insert a new technician into the database
2. Delete an existing airplane from the database
3. Update the expertise of an existing Technician
4. List Technician details for those whose salary is greater than the average
5. List Models that have experts, along with weight and capacity details
6. List total number of Technicians per Model
7. List the details of Tests for a specific plane, ordered by maximum score
8. List most recent Exam dates and Union Number for each Traffic Controller
9. List the number of tests done on a plane, ordered by Technicians
10. List information for tests done between SEP05 and DEC05, ordered by Test Number
4
| empID = 1| empName = Aaron| ssn = 1234| address = 1234 Street St. PA| phoneNum = 1234567890| salary = 2001| unionNum = 1000| techModNum = BC234
| empID = 2| empName = Chris| ssn = 2345| address = 1234 Street St. PA| phoneNum = 2345678901| salary = 2002| unionNum = 1000| techModNum = JK012
| empID = 4| empName = Alex| ssn = 4567| address = 1111 Drive Dr. FL| phoneNum = 4567890213| salary = 2000| unionNum = 1000| techModNum = CD345
| empID = 7| empName = Lindsey| ssn = 7890| address = 3333 Avenue St. MD| phoneNum = 7890123456| salary = 1999| unionNum = 1000| techModNum = FG678
| empID = 11| empName = Manvi| ssn = 1111| address = 4567 Avenue Av. HI| phoneNum = 1122334455| salary = 2999| unionNum = 1000| techModNum = DE456
| empID = 12| empName = Oriana| ssn = 1112| address = 4567 Avenue Av. HI| phoneNum = 5544332211| salary = 2999| unionNum = 1000| techModNum = HI890
Program ended with exit code: 0

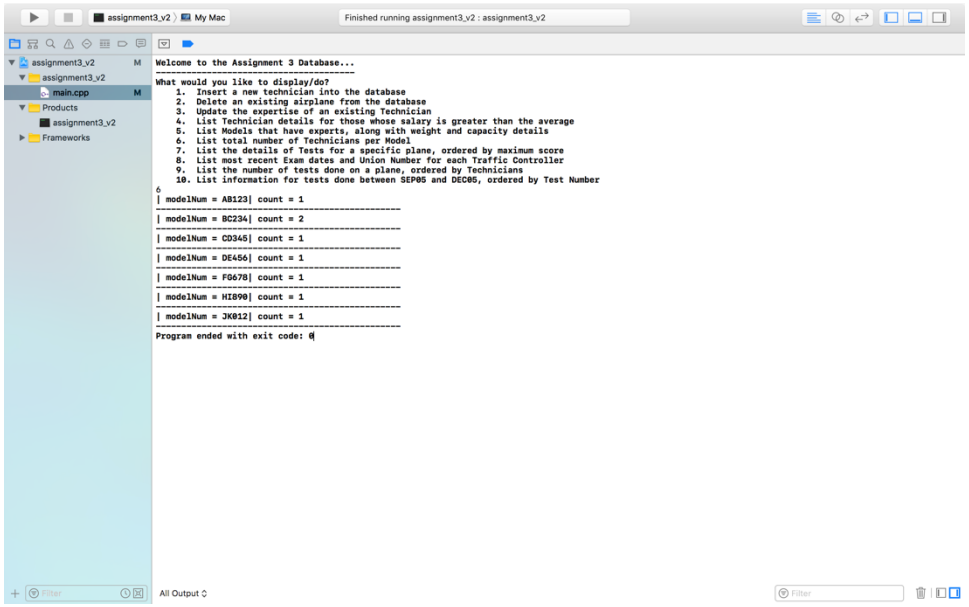

bte423
Airplane
PlaneTest
Employee
Technician
Employee
Limit to 1000 rows
1 SELECT empID, empName, ssn, address, phoneNum, salary, unionNum, techModNum
2 FROM `Employee`, `Technician`
3 WHERE salary > (SELECT AVG(salary) FROM `Employee`)
4 AND empID = techID;
100% 20:4
Result Grid
Filter Rows:
Search
Export:
empID empName ssn address phoneNum salary unionNum techModNum
1 Aaron 1234 1234 Street St. PA 1234567890 2001 1000 BC234
2 Chris 2345 1234 Street St. PA 2345678901 2002 1000 JK012
4 Alex 4567 1111 Drive Dr. FL 4567890213 2000 1000 CD345
7 Lindsey 7890 3333 Avenue St... 7890123456 1999 1000 FG678
11 Manvi 1111 4567 Avenue Av... 1122334455 2999 1000 DE456
12 Oriana 1112 4567 Avenue Av... 5544332211 2999 1000 HI890
Result 2
Read Only
Action Output
Time Action Response Duration / Fetch Time
20 22:26:08 SELECT testRegNum, testNum, testTechID, date, hours, score, testName, maxScore FROM `... 4 row(s) returned 0.044 sec / 0.000017...
21 22:26:15 SELECT * FROM final_project.Technician LIMIT 0, 1000 7 row(s) returned 0.044 sec / 0.00000...
22 22:28:13 SELECT * FROM final_project.Airplane LIMIT 0, 1000 18 row(s) returned 0.047 sec / 0.00004...
23 22:29:57 SELECT * FROM final_project.Employee LIMIT 0, 1000 14 row(s) returned 0.044 sec / 0.000013...
24 22:30:07 SELECT empID, empName, ssn, address, phoneNum, salary, unionNum, techModNum FROM... 6 row(s) returned 0.045 sec / 0.000016...
Query Completed
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
PERFORMANCE
Dashboard
Performance Reports
Performance Schema Setup
SCHEMAS
Filter objects
assignment2
final_project
Tables
Accommodations
Airplane
Employee
Model
PlaneTest
Technician
Test Type
TrafficController
Views
Stored Procedures
Functions
Result Grid
Form Editor
Field Types
Query Stats
Execution Plan


bte423
Airplane
PlaneTest
Employee
Technician
Employee
Limit to 1000 rows
1 SELECT AVG(salary) FROM Employee;
100% 33:1
Result Grid
Filter Rows:
Search
Export:
AVG(salary)
1953.5714
Result 3
Read Only
Action Output
Time Action Response Duration / Fetch Time
21 22:26:15 SELECT * FROM final_project.Technician LIMIT 0, 1000 7 row(s) returned 0.044 sec / 0.00000...
22 22:28:13 SELECT * FROM final_project.Airplane LIMIT 0, 1000 18 row(s) returned 0.047 sec / 0.00004...
23 22:29:57 SELECT * FROM final_project.Employee LIMIT 0, 1000 14 row(s) returned 0.044 sec / 0.000013...
24 22:30:07 SELECT empID, empName, ssn, address, phoneNum, salary, unionNum, techModNum FROM... 6 row(s) returned 0.045 sec / 0.000016...
25 22:30:45 SELECT AVG(salary) FROM Employee LIMIT 0, 1000 1 row(s) returned 0.044 sec / 0.000010...
Query Completed
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
PERFORMANCE
Dashboard
Performance Reports
Performance Schema Setup
SCHEMAS
Filter objects
assignment2
final_project
Tables
Accommodations
Airplane
Employee
Model
PlaneTest
Technician
Test Type
TrafficController
Views
Stored Procedures
Functions
Result Grid
Form Editor
Field Types
Query Stats
Execution Plan

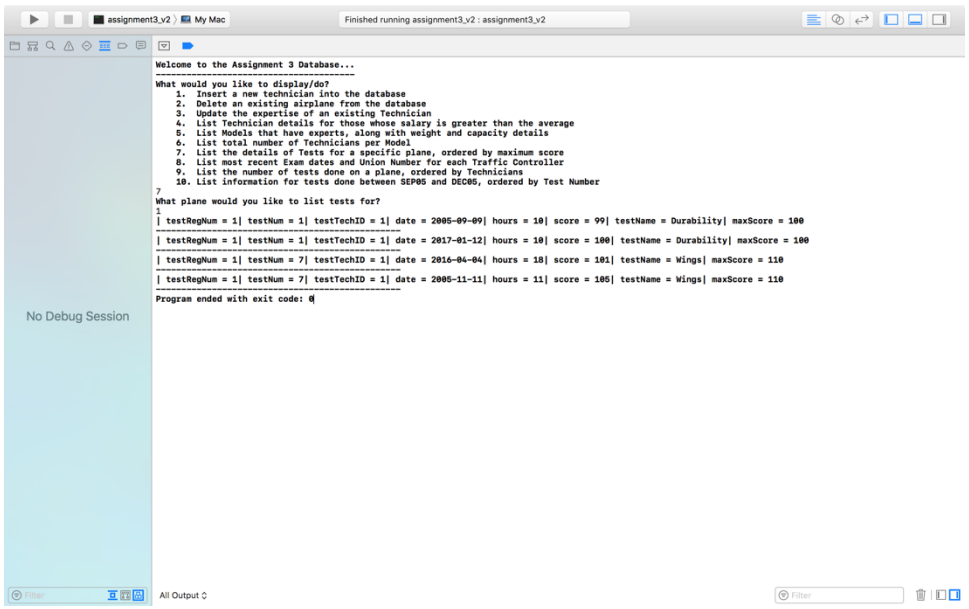
5.


assignment3_v2 〉 My Mac
Finished running assignment3_v2 : assignment3_v2
No Debug Session
All Output
Welcome to the Assignment 3 Database...
What would you like to display/do?
1. Insert a new technician into the database
2. Delete an existing airplane from the database
3. Update the expertise of an existing Technician
4. List Technician details for those whose salary is greater than the average
5. List Models that have experts, along with weight and capacity details
6. List total number of Technicians per Model
7. List the details of Tests for a specific plane, ordered by maximum score
8. List most recent Exam dates and Union Number for each Traffic Controller
9. List the number of tests done on a plane, ordered by Technicians
10. List information for tests done between SEP05 and DEC05, ordered by Test Number
5
| modelNum = AB123| modelCap = 50| modelWeight = 40
| modelNum = BC234| modelCap = 75| modelWeight = 60
| modelNum = BC234| modelCap = 75| modelWeight = 60
| modelNum = CD345| modelCap = 40| modelWeight = 30
| modelNum = DE456| modelCap = 55| modelWeight = 50
| modelNum = FG678| modelCap = 60| modelWeight = 56
| modelNum = HI890| modelCap = 50| modelWeight = 47
| modelNum = JK012| modelCap = 80| modelWeight = 66
Program ended with exit code: 0

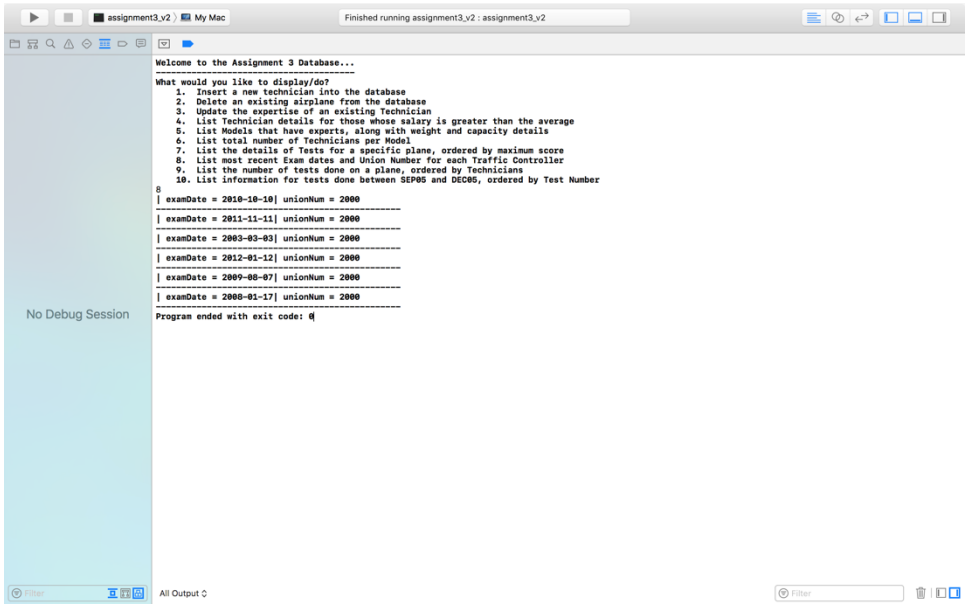
6.


assignment3_v2 〉 My Mac
Finished running assignment3_v2 : assignment3_v2
assignment3_v2
assignment3_v2
main.cpp
Products
assignment3_v2
Frameworks
All Output
Welcome to the Assignment 3 Database...
What would you like to display/do?
1. Insert a new technician into the database
2. Delete an existing airplane from the database
3. Update the expertise of an existing Technician
4. List Technician details for those whose salary is greater than the average
5. List Models that have experts, along with weight and capacity details
6. List total number of Technicians per Model
7. List the details of Tests for a specific plane, ordered by maximum score
8. List most recent Exam dates and Union Number for each Traffic Controller
9. List the number of tests done on a plane, ordered by Technicians
10. List information for tests done between SEP05 and DEC05, ordered by Test Number
6
| modelNum = AB123| count = 1
| modelNum = BC234| count = 2
| modelNum = CD345| count = 1
| modelNum = DE456| count = 1
| modelNum = FG678| count = 1
| modelNum = HI890| count = 1
| modelNum = JK012| count = 1
Program ended with exit code: 0

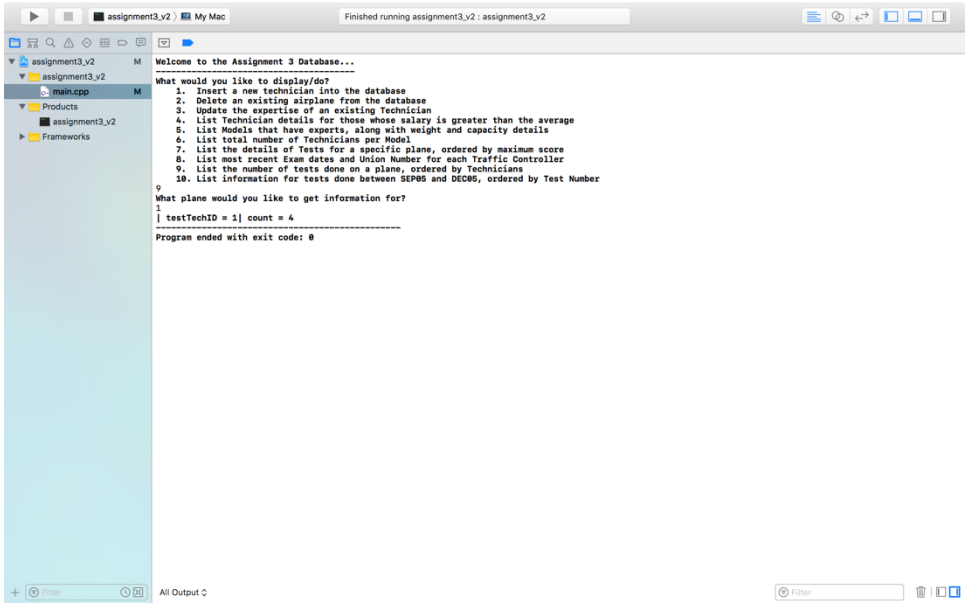
7.


assignment3_v2 〉 My Mac
Finished running assignment3_v2 : assignment3_v2
Welcome to the Assignment 3 Database...
What would you like to display/do?
1. Insert a new technician into the database
2. Delete an existing airplane from the database
3. Update the expertise of an existing Technician
4. List Technician details for those whose salary is greater than the average
5. List Models that have experts, along with weight and capacity details
6. List total number of Technicians per Model
7. List the details of Tests for a specific plane, ordered by maximum score
8. List most recent Exam dates and Union Number for each Traffic Controller
9. List the number of tests done on a plane, ordered by Technicians
10. List information for tests done between SEP05 and DEC05, ordered by Test Number
7
What plane would you like to list tests for?
1
| testRegNum = 1| testNum = 1| testTechID = 1| date = 2005-09-09| hours = 10| score = 99| testName = Durability| maxScore = 100
| testRegNum = 1| testNum = 1| testTechID = 1| date = 2017-01-12| hours = 10| score = 100| testName = Durability| maxScore = 100
| testRegNum = 1| testNum = 7| testTechID = 1| date = 2016-04-04| hours = 18| score = 101| testName = Wings| maxScore = 110
| testRegNum = 1| testNum = 7| testTechID = 1| date = 2005-11-11| hours = 11| score = 105| testName = Wings| maxScore = 110
Program ended with exit code: 0
No Debug Session
All Output

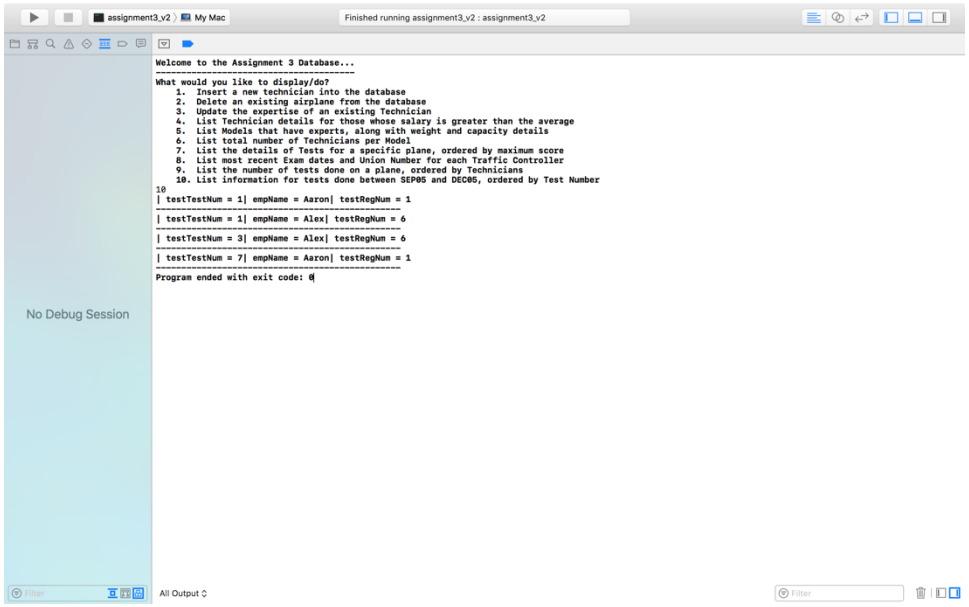
8.


assignment3_v2 〉 My Mac
Finished running assignment3_v2 : assignment3_v2
Welcome to the Assignment 3 Database...
What would you like to display/do?
1. Insert a new technician into the database
2. Delete an existing airplane from the database
3. Update the expertise of an existing Technician
4. List Technician details for those whose salary is greater than the average
5. List Models that have experts, along with weight and capacity details
6. List total number of Technicians per Model
7. List the details of Tests for a specific plane, ordered by maximum score
8. List most recent Exam dates and Union Number for each Traffic Controller
9. List the number of tests done on a plane, ordered by Technicians
10. List information for tests done between SEP05 and DEC05, ordered by Test Number
8
| examDate = 2010-10-10| unionNum = 2000
| examDate = 2011-11-11| unionNum = 2000
| examDate = 2003-03-03| unionNum = 2000
| examDate = 2012-01-12| unionNum = 2000
| examDate = 2009-08-07| unionNum = 2000
| examDate = 2008-01-17| unionNum = 2000
Program ended with exit code: 0
No Debug Session
All Output

9.


assignment3_v2 〉 My Mac
Finished running assignment3_v2 : assignment3_v2
assignment3_v2
assignment3_v2
main.cpp
Products
assignment3_v2
Frameworks
Welcome to the Assignment 3 Database...
What would you like to display/do?
1. Insert a new technician into the database
2. Delete an existing airplane from the database
3. Update the expertise of an existing Technician
4. List Technician details for those whose salary is greater than the average
5. List Models that have experts, along with weight and capacity details
6. List total number of Technicians per Model
7. List the details of Tests for a specific plane, ordered by maximum score
8. List most recent Exam dates and Union Number for each Traffic Controller
9. List the number of tests done on a plane, ordered by Technicians
10. List information for tests done between SEP05 and DEC05, ordered by Test Number
9
What plane would you like to get information for?
1
| testTechID = 1| count = 4
Program ended with exit code: 0
All Output

10.


assignment3_v2 〉 My Mac
Finished running assignment3_v2 : assignment3_v2
No Debug Session
Welcome to the Assignment 3 Database...
What would you like to display/do?
1. Insert a new technician into the database
2. Delete an existing airplane from the database
3. Update the expertise of an existing Technician
4. List Technician details for those whose salary is greater than the average
5. List Models that have experts, along with weight and capacity details
6. List total number of Technicians per Model
7. List the details of Tests for a specific plane, ordered by maximum score
8. List most recent Exam dates and Union Number for each Traffic Controller
9. List the number of tests done on a plane, ordered by Technicians
10. List information for tests done between SEP05 and DEC05, ordered by Test Number
10
| testTestNum = 1| empName = Aaron| testRegNum = 1
| testTestNum = 1| empName = Alex| testRegNum = 6
| testTestNum = 3| empName = Alex| testRegNum = 6
| testTestNum = 7| empName = Aaron| testRegNum = 1
Program ended with exit code: 0
All Output