

Alejandro Sanchez

BTE423

Final Project – Case Study: Airport Information System

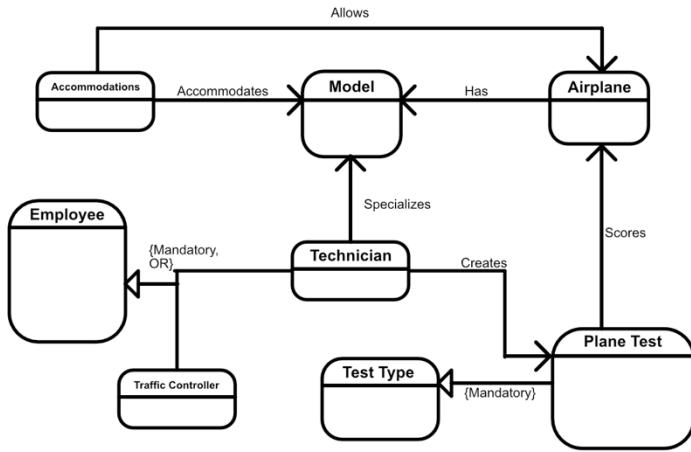
Construction of the Conceptual Data Model

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

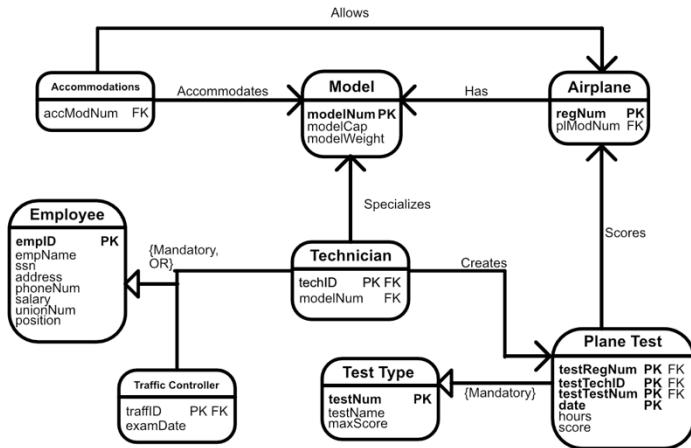
- i. All other attributes would not conform to the Entity Integrity Constraint, as all others but the Social Security Number can be shared amongst multiple employees
 - b. Test Type – Test Number
 - i. Assuming that the Test Number refers to a form of identification given to a given test type, rather 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:



- 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

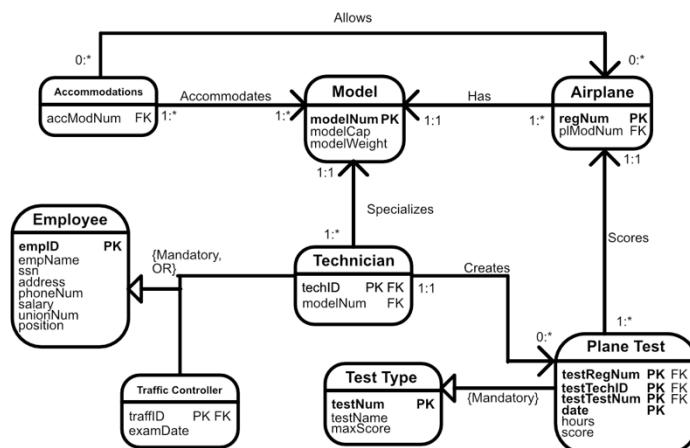
The ideal follow up to the Conceptual Data Model would be to begin designating attributes in the graphical representation, to identify how each entity relates to each other in terms of Relational Integrity and dependency, and at the same time it serves as a way to avoid data redundancy.



- Within this new graphical representation, all attributes have been added to what are to be considered the optimal entities.
 - Some tables do not require all information, so it would be best to exclude this information and store them separately.
 - 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.
- Some information has been distributed in order to make changes to information easier.
 - 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.
 - Both Primary Keys and Foreign Keys have been identified.
 - Foreign keys chosen are to avoid redundancy and/or errors.
 - 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.
 - Currently, potential constraints are defined as follows:
 - Given the information, an employee must have a position of Technician or Traffic Controller
 - 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
 - Each Airplane instance must only be of one Model, which must exist and be accommodated for by the airport
 - Each Model can only have one Capacity and one Weight
 - Each Test Type must have a name
 - Every employee must have a different SSN and ID
 - All other non-primary key attributes can be shared amongst various instances of entities.
 - No attributes seem to need the option to contain a NULL value
 - Some attributes may have default values:
 - MaxScore can default to 100
 - 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:

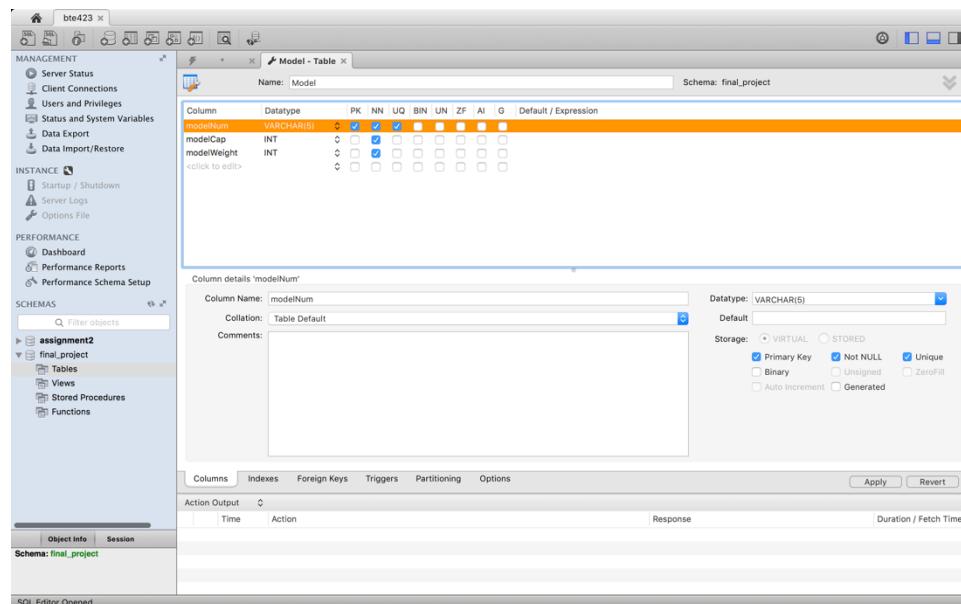
1. The Accommodations list accommodates one to many Models
2. Each Model is accommodated by one or more Accommodations lists (in this case study, there is only one list)
3. Each Airplane has only one model
4. Each model can belong to one to many Airplanes
5. Each Technician specializes in one and only 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,
```

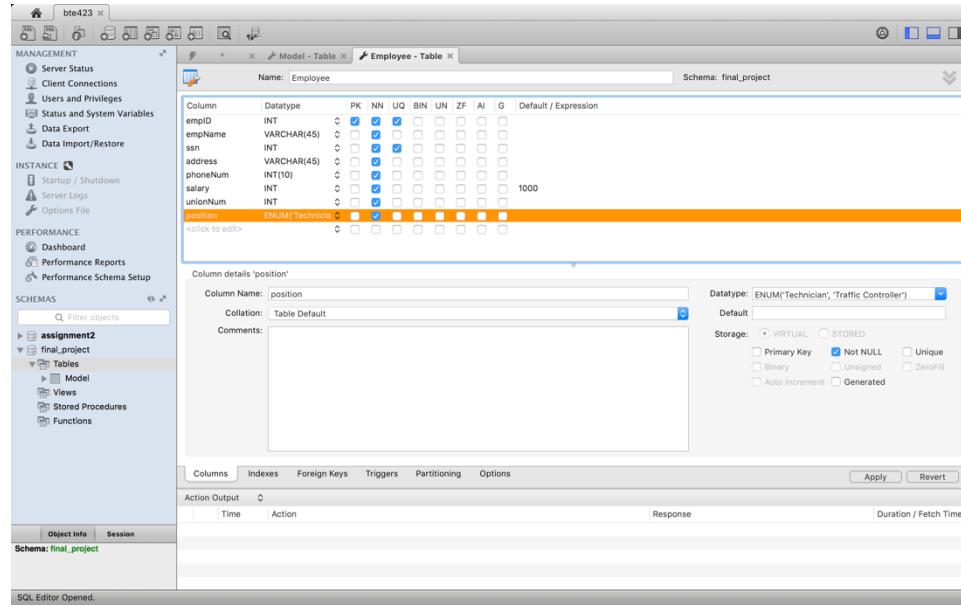
```

`modelWeight` int(11) NOT NULL,
PRIMARY KEY (`modelNum`),
UNIQUE KEY `modelNum_UNIQUE` (`modelNum`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

```

2. Employee:

a. Screenshot:



b. Query:

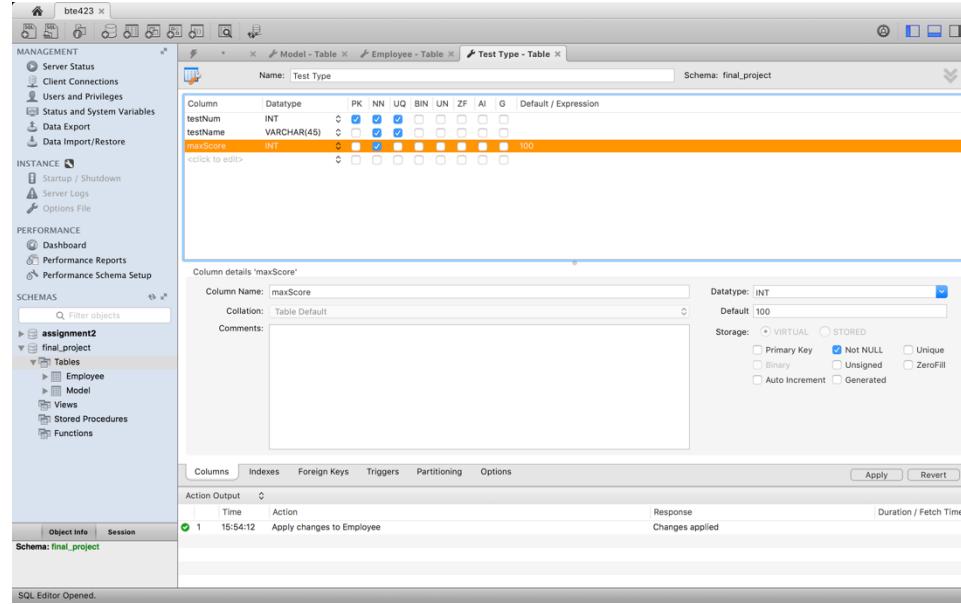
```

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:

The screenshot shows two panels of MySQL Workbench. The left panel displays the 'final_project' schema with various tables like Employee, Model, and Accommodations. The right panel is focused on the 'Accommodations - Table' configuration.

Table Configuration:

- Name:** Accommodations
- Schema:** final_project
- Columns:**
 - accomModNum (VARCHAR(5)) - Primary Key, Not Null, Unique
- Foreign Keys:**
 - modNum (final_project.Model) REFERENCES accomModNum

Action Output:

Time	Action	Response	Duration / Fetch Time
7	13:53:35 SELECT * FROM final_project.'Test Type' LIMIT 0, 1000	8 row(s) returned	0.044 sec / 0.00002...
8	14:01:47 SELECT * FROM final_project.TrafficController LIMIT 0, 1000	6 row(s) returned	0.045 sec / 0.000021...
9	14:30:06 SELECT * FROM final_project.Technician LIMIT 0, 1000	6 row(s) returned	0.044 sec / 0.000015...
10	16:54:58 SELECT * FROM final_project.Employee LIMIT 0, 1000	12 row(s) returned	0.045 sec / 0.00002...
11	16:57:17 SELECT * FROM final_project.Model LIMIT 0, 1000	25 row(s) returned	0.044 sec / 0.00002...

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:

The screenshot shows two instances of MySQL Workbench. The top instance displays the 'Airplane - Table' configuration. The 'Columns' tab lists 'regNum' (INT, PK, NN, UQ) and 'pModNum' (VARCHAR(5)). The 'Foreign Keys' tab shows a foreign key 'pModNum' referencing the 'Accommodations' table's 'pModNum' column. The bottom instance shows the 'Technician - Model - Table' configuration, where the 'Airplane' table is selected. The 'Foreign Keys' tab also shows the same foreign key relationship.

b. Query:

```

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

```

3. Technician:

a. Screenshots:

Table: Technician

Columns:

- techID int(11) PK
- techModNum varchar(5)

Action Output

Time	Action	Response	Duration / Fetch Time
1 16:05:50	Apply changes to Accommodations	Changes applied	
2 16:05:15	Apply changes to Airplane	Changes applied	
3 16:19:00	Apply changes to Technician	Changes applied	

Table: Technician

Foreign Key

Foreign Key	Referenced Table
techID	'final_project'.'Employee'
techModNum	'final_project'.'Model'

Action Output

Time	Action	Response	Duration / Fetch Time
7 13:53:35	SELECT * FROM final_project.'Test Type' LIMIT 0, 1000	8 row(s) returned	0.044 sec / 0.00002...
8 14:04:47	SELECT * FROM final_project.TrafficController LIMIT 0, 1000	6 row(s) returned	0.045 sec / 0.000021...
9 14:05:05	SELECT * FROM final_project.Technician LIMIT 0, 1000	6 row(s) returned	0.044 sec / 0.000015...
10 16:04:58	SELECT * FROM final_project.Employee LIMIT 0, 1000	12 row(s) returned	0.045 sec / 0.00002...
11 16:57:17	SELECT * FROM final_project.Model LIMIT 0, 1000	25 row(s) returned	0.044 sec / 0.00002...

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:

The screenshot shows the MySQL Workbench interface for creating a table named 'TrafficController'. The 'examDate' column is currently selected. The column details pane shows the column name 'examDate' with a datatype of 'DATE'. Other options include 'Default', 'Storage', and various check boxes for constraints like Primary Key, Not NULL, Unique, Binary, Unsigned, Auto Increment, and Generated.

The screenshot shows the configuration of a foreign key named 'traffID' for the 'TrafficController' table. It points to the 'Employee' table with the 'empID' column as the referenced column. The foreign key details pane shows the column 'traffID' and the referenced column 'empID' with the 'On Update' and 'On Delete' actions both set to 'NO ACTION'.

b. Query:

```

CREATE TABLE `final_project`.`TrafficController` (
`traffID` INT(11) NOT NULL,

```

```

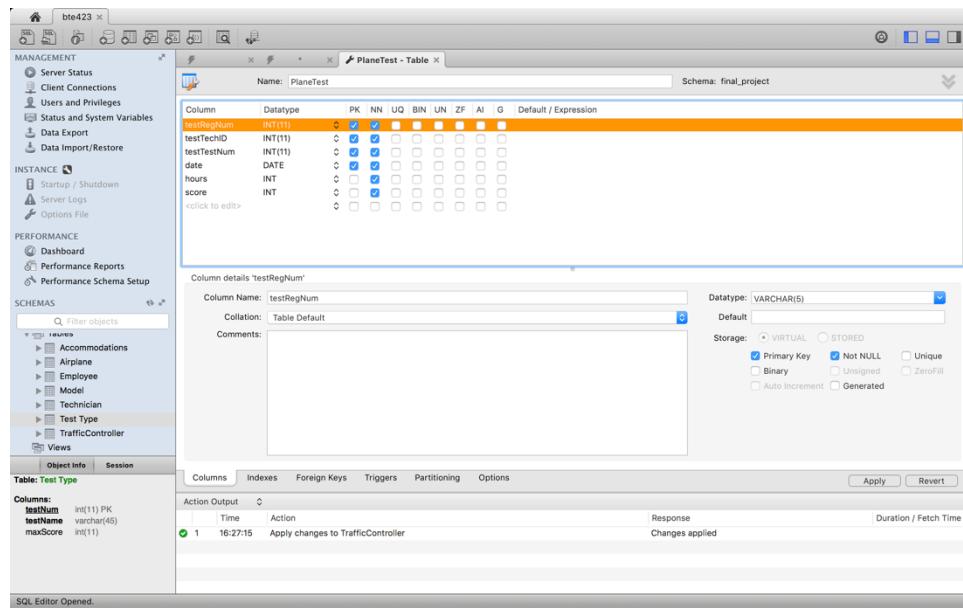
`examDate` DATE NOT NULL,
PRIMARY KEY (`trafficID`),
UNIQUE INDEX `trafficID_UNIQUE` (`trafficID` ASC),
CONSTRAINT `trafficID`
FOREIGN KEY (`trafficID`)
REFERENCES `final_project`.`Employee` (`empID`)
ON DELETE NO ACTION
ON UPDATE NO ACTION);

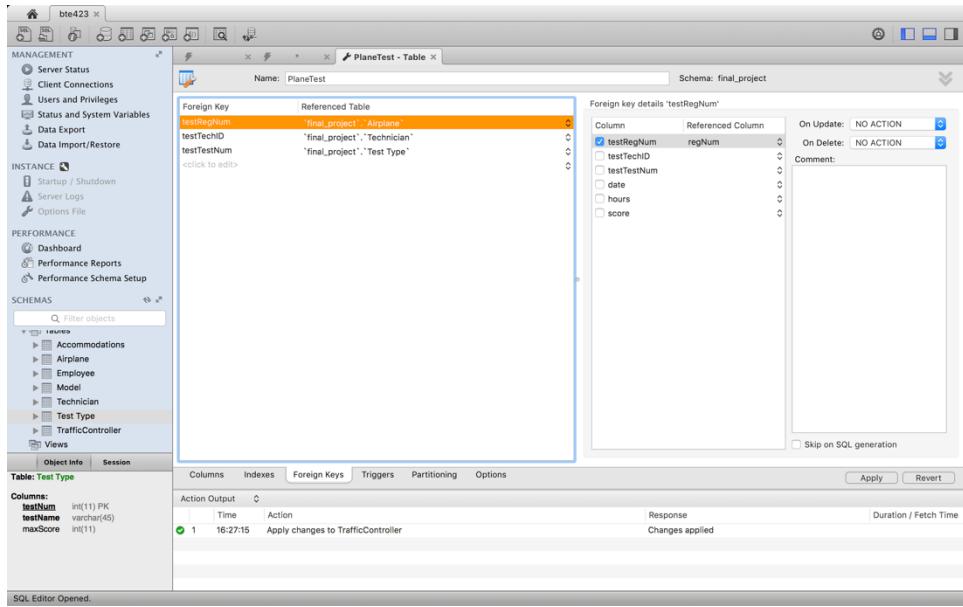
```

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

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

The screenshot shows the MySQL Workbench interface with the 'Model' table selected. The data grid displays 25 rows of model information. A query result grid below shows the execution of a SELECT statement on the Model table.

modelNum	modelCap	modelWeight
AB123	50	40
BC234	75	60
CD345	40	30
DE456	55	50
EF567	45	28
FG678	60	56
GH789	65	61
HI890	50	47
IJ901	75	55
JK012	80	66
KL123	70	55
LM234	60	59
MN345	80	90
NO456	45	30
OP567	40	35
PO678	65	59

Query Results:

Action Output	Time	Action	Response	Duration / Fetch Time
1	17:58:53	SELECT * FROM final_project.Model LIMIT 0, 1000	0 row(s) returned	0.04 sec / 0.000007...

- 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:

The screenshot shows the MySQL Workbench interface with the 'Employee' table selected. The left sidebar displays the database schema with the 'Employee' table highlighted. The main area shows a grid of 12 employee records with columns: empID, empName, ssn, address, phoneNum, salary, unionNum, and position. The 'position' column indicates 6 employees as 'Traffic Controller' and 6 as 'Technician'. The bottom section shows the SQL query history for inserting and selecting data from the Employee table.

empID	empName	ssn	address	phoneNum	salary	unionNum	position
1	Aaron	1234	1234 Street St. PA	1234567890	2001	1000	Technician
2	Chris	2345	1234 Street St. PA	2345678901	2002	1000	Technician
3	Sebastian	3456	1234 Street St. PA	3456789012	1999	2000	Traffic Controller
4	Alex	4567	1111 Drive Dr. PA	4567890213	2000	1000	Technician
5	Emily	5678	1001 Avenue W...	5678901234	1850	2000	Traffic Controller
6	Sophia	6789	1001 Avenue W...	6789012345	1840	2000	Traffic Controller
7	Lindsey	7890	3333 Avenue SL...	7890123456	1830	2000	Technician
8	Joshua	8901	2945 Drive Av. FL	8901234567	1889	2000	Traffic Controller
9	Nick	9012	0056 Street Dr. FL	9012345678	1800	2000	Traffic Controller
10	Parth	123	3333 Avenue SL...	0123456789	1750	2000	Traffic Controller
11	Mari	1111	4567 Avenue Av...	112334456	2999	1000	Technician
12	Oriana	1112	4567 Avenue Av...	5544332111	2999	1000	Technician

Action Output:

Action	Time	Response	Duration / Fetch Time
60	20:43:22	INSERT INTO `final_project`.`Employee` (`empID`, `empName`, `ssn`, `address`, `phoneNum`, `salary`, `unionNum`)	1048: Column 'salary' cannot be null
61	20:43:23	INSERT INTO `final_project`.`Employee` (`empID`, `empName`, `ssn`, `address`, `phoneNum`, `salary`, `unionNum`)	1048: Column 'salary' cannot be null
62	20:43:43	Apply changes to Employee	Changes applied
63	20:45:56	SELECT * FROM final_project.Employee LIMIT 0, 1000	12 row(s) returned
64	20:47:01	SELECT * FROM final_project.Employee WHERE `salary` = 900 LIMIT 0, 1000	0 row(s) returned

3. Test Type

- a. 8 initial test types were generated.

The screenshot shows the MySQL Workbench interface with a query editor window titled 'Employee' and a results grid window titled 'Test Type'. The results grid displays the following data:

testNum	testName	maxScore
1	Durability	100
2	Speed	120
3	Safety	100
4	Weight	100
5	Overall	100
6	Length	105
7	Wings	110
8	Engine	100

Below the results grid, the 'Action Output' pane shows the executed SQL query:

```
1 21:00:04 SELECT * FROM final_project.Test Type LIMIT 0, 1000
```

4. Accommodations

- 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.

The screenshot shows the MySQL Workbench interface with a query editor window titled 'Employee' and a results grid window titled 'Accommodations'. The results grid displays the following data:

accModNum
BC123
BC234
CD345
DE456
EF567
FG678
GH789
HI890
IJ901
JK012

Below the results grid, the 'Action Output' pane shows the executed SQL query:

```
1 21:00:04 SELECT * FROM final_project.Test Type LIMIT 0, 1000
2 22:16:27 SELECT * FROM final_project.Test Type LIMIT 0, 1000
3 22:17:39 SELECT * FROM final_project.Accommodations LIMIT 0, 1000
4 22:21:45 SELECT * FROM final_project.Airplane LIMIT 0, 1000
5 22:21:54 SELECT * FROM final_project.Model LIMIT 0, 1000
```

5. Airplane

- 2 airplanes were created for each model accommodated by the airport

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
1 • SELECT * FROM final_project.Airplane;
```

The results grid displays the following data:

regNum	pModNum
1	AB120
2	AB123
3	BC234
4	BC234
5	CD245
6	CD245
7	DE456
8	DE456
9	EF567
10	EF567
11	FG678
12	FG678
13	GH789
14	GH789
15	H890
16	H890
17	JK012
18	JK012
19	JK012
20	JK012
21	JK012
22	JK012

Below the results grid, the "Action Output" section shows the execution history:

Action	Time	Response	Duration / Fetch Time
1 21:00:04	SELECT * FROM final_project.Test Type LIMIT 0, 1000	0 row(s) returned	0.047 sec / 0.00001...
2 22:16:27	SELECT * FROM final_project.Test Type LIMIT 0, 1000	8 row(s) returned	0.044 sec / 0.00000...
3 22:13:39	SELECT * FROM final_project.Accommodations LIMIT 0, 1000	0 row(s) returned	0.044 sec / 0.00000...
4 22:14:45	SELECT * FROM final_project.Airplane LIMIT 0, 1000	0 row(s) returned	0.044 sec / 0.000010...
5 22:15:54	SELECT * FROM final_project.Model LIMIT 0, 1000	25 row(s) returned	0.044 sec / 0.00002...

6. Technician

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

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
1 • SELECT * FROM final_project.Technician;
```

The results grid displays the following data:

techID	techModNum
1	AB123
2	JK012
4	CD245
7	FG678
11	DE456
12	H890
21	JK012

Below the results grid, the "Action Output" section shows the execution history:

Action	Time	Response	Duration / Fetch Time
1 13:29:50	SELECT * FROM final_project.Technician LIMIT 0, 1000	0 row(s) returned	0.044 sec / 0.000014...
2 13:30:40	SELECT * FROM final_project.Employee LIMIT 0, 1000	12 row(s) returned	0.044 sec / 0.00002...
3 13:31:41	SELECT * FROM final_project.Accommodations LIMIT 0, 1000	10 row(s) returned	0.044 sec / 0.000014...

7. Traffic Controller

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

Query Results:

trafficID	examDate
3	2010-10-...
5	2011-11-11
6	2005-03-...
8	2012-01-...
9	2008-08-...
10	2009-01-...
	MULL
	MULL

Action History:

Time	Action	Response	Duration / Fetch Time
15:29:50	SELECT * FROM final_project.TrafficController LIMIT 0, 1000	0 row(s) returned	0.044 sec / 0.000014...
15:30:40	SELECT * FROM final_project.Employee LIMIT 0, 1000	12 row(s) returned	0.044 sec / 0.00002...
15:31:41	SELECT * FROM final_project.Accommodations LIMIT 0, 1000	10 row(s) returned	0.044 sec / 0.000014...
15:34:03	SELECT * FROM final_project.TrafficController LIMIT 0, 1000	0 row(s) returned	0.045 sec / 0.000015...

8. Plane Test

- Two tests were logged for each plane recorded at the airport.

Query Results:

testRegNum	testTechID	testTestNum	date	hours	score
1	1	1	2017-01-12	10	100
1	1	7	2016-04-04	18	101
5	11	4	2013-11-25	82	
6	4	3	2015-01-12	5	75
6	4	3	2015-01-12	12	90
11	2	2	2009-09-09	11	23
11	7	5	2009-09-09	11	23
16	12	6	2007-07-07	29	104
16	12	4	2004-05-06	8	95
11	7	3	2003-02-01	13	93
6	4	1	2002-02-02	23	19
7	11	2	2001-02-03	19	114
19	2	8	2001-01-01	2	99

Action History:

Time	Action	Response	Duration / Fetch Time
13:34:03	SELECT * FROM final_project.TrafficController LIMIT 0, 1000	0 row(s) returned	0.045 sec / 0.000015...
13:32:56	SELECT * FROM final_project.PlaneTest LIMIT 0, 1000	0 row(s) returned	0.133 sec / 0.000008...
13:32:23	SELECT * FROM final_project.Airplane LIMIT 0, 1000	20 row(s) returned	0.044 sec / 0.000017...
13:31:35	SELECT * FROM final_project.Test_Type LIMIT 0, 1000	8 row(s) returned	0.044 sec / 0.00002...
14:01:47	SELECT * FROM final_project.TrafficController LIMIT 0, 1000	6 row(s) returned	0.046 sec / 0.000021...

Sample Queries

- Two queries must be executed in order to create a Technician, one to create the employee and another to assign their expertise.
 - `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');`
 - `INSERT INTO `final_project`.`Technician`(`techID`, `techModNum`) VALUES ('13', 'AB123');`
- `DELETE FROM `final_project`.`Airplane` WHERE `regNum`='20';`
- `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;
```

The screenshot shows the MySQL Workbench interface with the following details:

- SQL Editor:** Contains the query:


```
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;
```
- Result Grid:** Displays the results of the query, showing two rows of data:

empID	empName	ssn	address	phoneNum	salary	unionNum	techModNum
11	Manu	1111	4567 Avenue Av...	1122334455	2999	1000	DE456
12	Orana	1112	4567 Avenue Av...	5544332211	2999	1000	H1890
- Timeline:** Shows the execution steps and their durations:

Action	Time	Response	Duration / Fetch
19 16:05:35	SELECT * FROM final_project.Technician LIMIT 0, 1000	7 row(s) returned	0.044 sec / 0.0
20 16:05:50	SELECT * FROM final_project.PlanetTest LIMIT 0, 1000	7 row(s) returned	0.044 sec / 0.0
21 16:10:20	SELECT empID, empName, ssn, address, phoneNumber, salary, unionNumber, techModNum...	Error Code: 1054. Unknown column 'phoneNumber' in 'field list'	0.046 sec
22 16:10:59	SELECT empID, empName, ssn, address, phoneNumber, salary, unionNumber, techModNum...	Error Code: 1054. Unknown column 'phoneNumber' in 'field list'	0.171 sec
23 16:11:29	SELECT empID, empName, ssn, address, phoneNum, salary, unionNum, techModNum FROM...	2 row(s) returned	0.044 sec / 0.0
24 16:15:26	SELECT AVG(salary) FROM `Employee` LIMIT 0, 1000	1 row(s) returned	0.044 sec / 0.0

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

The screenshot shows the MySQL Workbench interface with the following details:

- SQL Editor:** Contains the query:


```
1 SELECT AVG(salary) FROM `Employee`;
```
- Result Grid:** Displays the results of the query, showing the average salary:

AVG(salary)
2026.8482
- Timeline:** Shows the execution steps and their durations:

Action	Time	Response	Duration / Fetch
20 16:05:50	SELECT * FROM final_project.PlanetTest LIMIT 0, 1000	12 row(s) returned	0.044 sec / 0.0
21 16:10:20	SELECT empID, empName, ssn, address, phoneNumber, salary, unionNumber, techModNum...	Error Code: 1054. Unknown column 'phoneNumber' in 'field list'	0.046 sec
22 16:10:59	SELECT empID, empName, ssn, address, phoneNumber, salary, unionNumber, techModNum...	Error Code: 1054. Unknown column 'phoneNumber' in 'field list'	0.171 sec
23 16:11:29	SELECT empID, empName, ssn, address, phoneNum, salary, unionNum, techModNum FROM...	2 row(s) returned	0.044 sec / 0.0
24 16:15:26	SELECT AVG(salary) FROM `Employee` LIMIT 0, 1000	1 row(s) returned	0.044 sec / 0.0

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;`

```

SELECT modelNum, modelCap, modelWeight
FROM `Model`, `Technician`
WHERE modelNum = techModNum;

```

modelNum	modelCap	modelWeight
AB123	50	40
BC234	75	60
CD345	40	30
DE456	65	55
FG678	60	55
HJ890	50	47
JK012	80	66

6. `SELECT modelNum, COUNT(techID)
FROM `Model`, `Technician`
WHERE modelNum = techModNum
GROUP BY modelNum;`

```

SELECT modelNum, COUNT(techID)
FROM `Model`, `Technician`
WHERE modelNum = techModNum
GROUP BY modelNum;

```

modelNum	COUNT(techID)
AB123	1
BC234	1
CD345	1
DE456	1
FG678	1
HJ890	1
JK012	1

7. `SELECT testRegNum, testName, testTechID, date, hours, score, 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.

```

SELECT testRegNum, testNum, testTechID, date, hours, score, testName, maxScore
FROM `PlaneTest`
WHERE testNum = testTestNum
AND testRegNum = 1
ORDER BY maxScore;

```

testRegNum	testNum	testTechID	date	hours	score	testName	maxScore
1	1	1	2005-09-09	10	99	Durability	100
1	1	1	2017-01-12	10	100	Durability	100
1	7	1	2016-04-04	18	101	Wings	110
1	7	1	2005-11-11	11	105	Wings	110
2	1	1	2016-04-04	18	101	Wings	110
2	1	1	2017-01-12	10	100	Durability	100
2	7	1	2005-11-11	11	105	Wings	110
3	1	1	2016-04-04	18	101	Wings	110
3	1	1	2017-01-12	10	100	Durability	100
3	7	1	2005-11-11	11	105	Wings	110

Action	Time	Action	Response	Duration / Fetch Time
13	21:09:50	SELECT * FROM final_project.Techician LIMIT 0, 1000	8 row(s) returned	0.044 sec / 0.000914...
14	21:13:31	SELECT * FROM final_project.Airplane LIMIT 0, 1000	19 row(s) returned	0.044 sec / 0.000911...
15	21:40:09	SELECT * FROM final_project.PlaneTest LIMIT 0, 1000	18 row(s) returned	0.044 sec / 0.000917...
16	21:54:27	SELECT * FROM final_project.Technician LIMIT 0, 1000	7 row(s) returned	0.046 sec / 0.000900...
17	22:11:38	SELECT testRegNum, testNum, testTechID, date, hours, score, testName, maxScore FROM `PlaneTest` WHERE testNum = testTestNum AND testRegNum = 1 ORDER BY maxScore;	4 row(s) returned	0.044 sec / 0.000902...

8. SELECT examDate, unionNum

```

FROM `Employee`, `TrafficController`
```

```

WHERE traffID = empID;
```

```

SELECT examDate, unionNum
FROM `Employee`, `TrafficController`
WHERE traffID = empID;

```

examDate	unionNum
2010-10-10	2000
2011-11-11	2000
2003-03-03	2000
2012-01-03	2000
2009-08-07	2000
2008-01-17	2000

Action	Time	Action	Response	Duration / Fetch Time
31	16:29:22	SELECT * FROM final_project.PlaneTest LIMIT 0, 1000	12 row(s) returned	0.044 sec / 0.0
32	16:31:13	SELECT * FROM `PlaneTest`, `Test Type` WHERE testNum = testTestNum AND testNum = 1...	Error Code: 1146, Table 'final_project.Plane Test' doesn't exist	0.046 sec
33	16:31:21	SELECT * FROM `PlaneTest`, `Test Type` WHERE testNum = testTestNum AND testNum = 1...	2 row(s) returned	0.044 sec / 0.0
34	16:32:12	SELECT testRegNum, testNum, testTechID, date, hours, score, testName, maxScore FROM `PlaneTest` WHERE testNum = testTestNum AND testRegNum = 1...	2 row(s) returned	0.044 sec / 0.0
35	16:35:20	SELECT examDate, unionNum FROM `Employee`, `TrafficController` WHERE traffID = empID...	6 row(s) returned	0.044 sec / 0.0

9. SELECT testTechID, COUNT(*)

```

FROM `PlaneTest`

WHERE testRegNum = 1

GROUP BY testTechID;

```

The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure, including the schema `final_project` and its tables: `Accommodations`, `Airplane`, `Employee`, `Model`, `PlaneTest`, `Technician`, `Test Type`, and `TrafficController`. The `PlaneTest` table is selected. The middle pane shows the query editor with the provided SQL code. The results grid on the right shows the output of the query:

testTechID	COUNT(*)
1	2

Below the results, the Action Output pane shows the execution log with six entries, all completed successfully.

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.

The screenshot shows the SQL Server Management Studio interface. The left pane shows the database structure with the `final_project` schema selected. The middle pane shows the query editor with the query: `SELECT * FROM final_project.PlaneTest;`. The results grid on the right shows 16 rows of data from the `PlaneTest` table. Below the results, the Action Output pane shows the execution log with six entries, all completed successfully.

b. `SELECT testTestNum, empName, testRegNum`

`FROM 'PlaneTest', 'Employee'`

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

```
AND empID = testTechID
```

```
ORDER BY testTestNum;
```

The screenshot shows the Oracle SQL Developer interface. On the left, the navigation pane displays 'MANAGEMENT' (Server Status, Client Connections, Users and Privileges, Status and System Variables, Data Export, Data Import/Restore), 'INSTANCE' (Startup / Shutdown, Server Logs, Options File), and 'PERFORMANCE' (Dashboard, Performance Reports, Performance Schema Setup). The 'SCHEMAS' section shows 'assignment2' and 'final_project' schemas, with 'final_project' expanded to show Tables, Views, Stored Procedures, and Functions. The central area contains a SQL editor window titled 'SQL File 5*' with the following query:

```
1 SELECT testTestNum, empName, testRegNum
  FROM 'PlaneTest'. 'Employee'
 WHERE date BETWEEN '2005-09-01' AND '2005-12-31'
   AND empID = testTechID
 ORDER BY testTestNum;
```

Below the editor is a 'Result Grid' showing the query results:

testTestNum	empName	testRegNum
1	Aaron	1
1	Alex	6
3	Alex	6
7	Aaron	1

At the bottom of the interface, a status bar indicates 'Query Completed'.

Console Sample Queries:

1.

The screenshot shows a terminal window titled 'assignment3_v2 : My Mac' with the message 'Finished running assignment3_v2 : assignment3_v2'. The window displays the following output:

```
Welcome to the Assignment 3 Database...
What would you like to 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 all models that have a salary for which the 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 expertise for a specific plane, along with maximum score
8. List recent Exam dates, location, 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 DEC06, ordered by Test Number

1. Please insert information as requested
ID: 14
Name: C++
ssn: 2226
address:
Street: 
phoneNum: 9999999999
salary: 1000
unionNum: 
expertise: A8123
Technician added...
| empID = 1| empName = Aaron| ssn = 1234| address = 1234 Street St. PA| phoneNum = 1234567890| salary = 2000| unionNum = 1000| techModNum = AB123
| empID = 14| empName = C++| ssn = 2226| address = Street| phoneNum = 9999999999| salary = 1000| unionNum = 1000| techModNum = A8123
| empID = 13| empName = New| ssn = 9876| address = 3333 Avenue St. MD| phoneNum = 9999999999| salary = 1111| unionNum = 1000| techModNum = BC234
| empID = 4| empName = Alex| ssn = 4567| address = 111 Drive Dr. FL| phoneNum = 4567890213| salary = 2000| unionNum = 1000| techModNum = CD345
| empID = 11| empName = Manvi| ssn = 1111| address = 4567 Avenue Av. HI| phoneNum = 1122334455| salary = 2999| unionNum = 1000| techModNum = DE456
| empID = 7| empName = Lindsey| ssn = 7698| address = 3333 Avenue St. MD| phoneNum = 7890123456| salary = 1999| unionNum = 1000| techModNum = FG678
| empID = 12| empName = Oriana| ssn = 1112| address = 4567 Avenue Av. HI| phoneNum = 5544332211| salary = 2999| unionNum = 1000| techModNum = HI569
| empID = 2| empName = Chris| ssn = 2345| address = 1234 Street St. PA| phoneNum = 2345678901| salary = 2002| unionNum = 1000| techModNum = JK012
Program ended with exit code: 0
```

2.

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 all models whose average test score 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 all tests done on 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 DEC06, ordered by Test Number

2 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

No Debug Session

All Output

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

PlaneTest

Employee

Technician

SELECT * FROM final_project.Airplane;

Result Grid

regNum	plnModNum
1	AB123
2	AB123
3	BC234
4	BC234
5	CD345
6	CD345
7	DE456
8	DE456
9	EF567
10	EF567
11	FG678
12	FG678
13	GH789
14	GH789
15	H890
16	H890
17	U501
19	JK012

Action Output

Action	Time	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 testLegNum, testNum, testTechID, date, hours, score, testName, maxScore FROM final_project.TechTest		4 row(s) returned	0.044 sec / 0.000017...
21 22:26:15 SELECT * FROM final_project.Techician 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
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 Models that have experts, along with weight and capacity details
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 DEC06, 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

No Debug Session
```

4.

```
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 DEC06, ordered by Test Number
4
| empID = 1| empName = Aaron| ssn = 1234| address = 1234 Street St. PA| phoneNum = 1234567890| salary = 2000| 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

No Debug Session
```

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

Query Completed

```

SELECT empID, empName, ssn, address, phoneNum, salary, unionNum, techModNum
FROM `Employee` , `Technician`
WHERE salary > (SELECT AVG(salary) FROM `Employee`)
AND empID = techID;
    
```

Result Grid

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	4567890123	2000	1000	CD345
7	Larry	7890	3333 Street St. PA	3333444455	2000	1000	EF789
11	Marni	1111	4567 Avenue Av...	1122334455	2999	1000	DE456
12	Oriana	1112	4567 Avenue Av...	5544332211	2999	1000	H890

Result 2

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 / 0.0000...
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...

Read Only

Query Completed

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

Query Completed

```

SELECT AVG(salary) FROM Employee;
    
```

Result Grid

AVG(salary)
1953.5714

Result 3

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 / 0.0000...
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...

Read Only

Query Completed

5.

```
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 the details of Tests 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 DEC06, 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 = JK901| modelCap = 80| modelWeight = 66
Program ended with exit code: 0
```

6.

```
assignment3.v2 : My Mac Finished running assignment3.v2 : assignment3.v2
assignment3.v2 M Welcome to the Assignment 3 Database...
assignment3.v2 M What would you like to display/do?
main.cpp M 1. Insert a new technician into the database
Products M 2. Delete an existing airplane from the database
assignment3.v2 M 3. Update the expertise of an existing Technician
Frameworks M 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 DEC06, 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 = JK901| 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 the details of Tests for a specific plane, ordered by maximum score
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 DEC06, 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 = 10| 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
```

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 DEC06, 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 = 2000-01-17| unionNum = 2000
Program ended with exit code: 0
```

9.

```
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 Technicians 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 DEC06, ordered by Test Number

? What plane would you like to get information for?
1 | testTechID = 1| count = 4
-----
Program ended with exit code: 0
```

10.

```
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 the details of Tests for a specific plane, ordered by maximum score
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 DEC06, 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
```