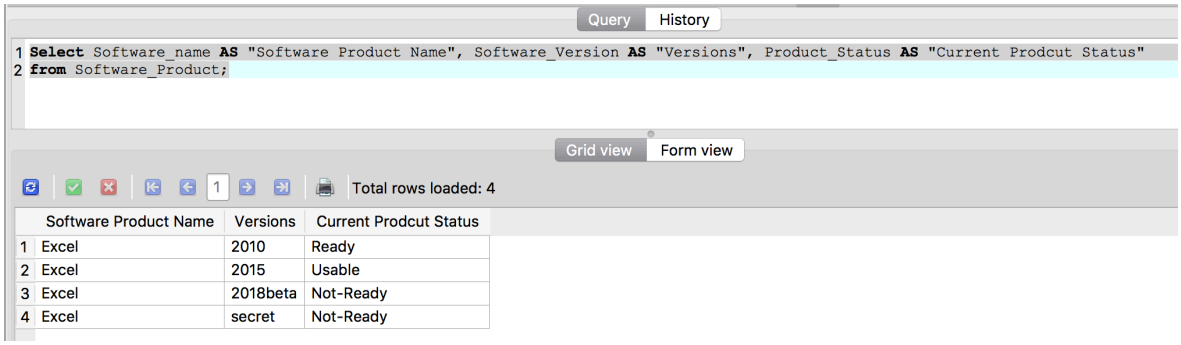
Somya Singh CSE7330 Fall 2017
Email: somyas@smu.edu
Student ID: 47304053 (Student ID)

Term Project Phase 3

1. List all software product names and versions and current product status.

   Select Software_name AS "Software Product Name", Software_Version AS "Versions",
   Product_Status AS "Current Prodcut Status"
   from Software_Product;



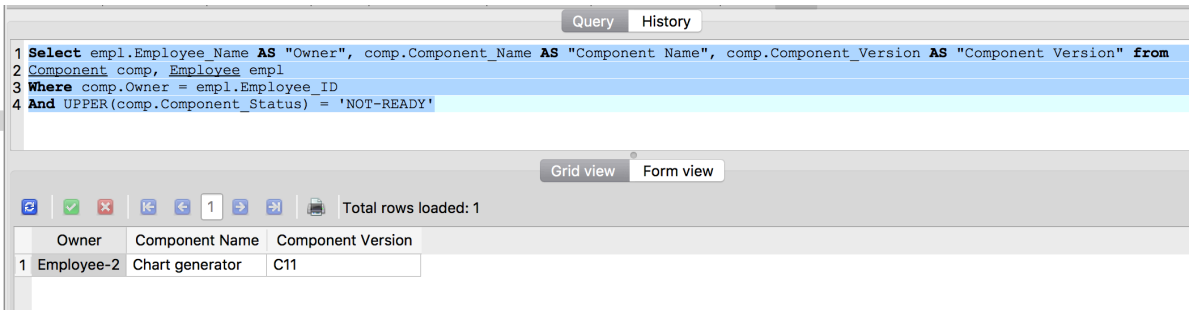
```
Select Software_name AS "Software Product Name", Software_Version AS "Versions", Product_Status AS "Current Prodcut Status"
from Software_Product;
```

Total rows loaded: 4

	Software Product Name	Versions	Current Prodcut Status
1	Excel	2010	Ready
2	Excel	2015	Usable
3	Excel	2018beta	Not-Ready
4	Excel	secret	Not-Ready

2. List the owner name, component name & version of all “not ready” components.

   Select empl.Employee_Name AS "Owner", comp.Component_Name AS "Component Name",
   comp.Component_Version AS "Component Version" from
   Component comp, Employee empl
   Where comp.Owner = empl.Employee_ID
   And UPPER(comp.Component_Status) = 'NOT-READY'



```
Select empl.Employee_Name AS "Owner", comp.Component_Name AS "Component Name", comp.Component_Version AS "Component Version" from
Component comp, Employee empl
Where comp.Owner = empl.Employee_ID
And UPPER(comp.Component_Status) = 'NOT-READY'
```

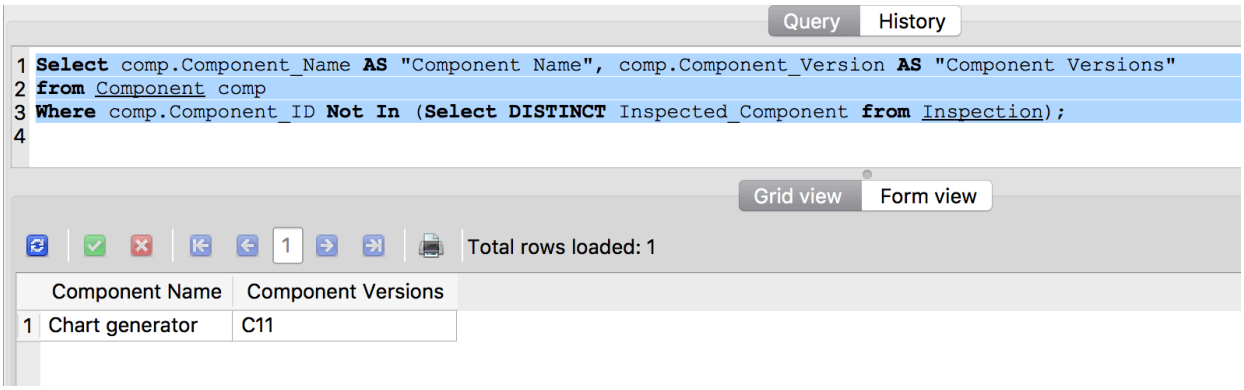
Total rows loaded: 1

	Owner	Component Name	Component Version
1	Employee-2	Chart generator	C11

Somya Singh
Email: somyas@smu.edu
Student ID: 47304053 (Student ID)

3. List all component names and versions that have not been inspected.

```
Select comp.Component_Name AS "Component Name", comp.Component_Version AS
"Component Versions"
from Component comp
Where comp.Component_ID Not In (Select DISTINCT Inspected_Component from Inspection);
```



Query | History

```
1 Select comp.Component_Name AS "Component Name", comp.Component_Version AS "Component Versions"
2 from Component comp
3 Where comp.Component_ID Not In (Select DISTINCT Inspected_Component from Inspection);
4
```

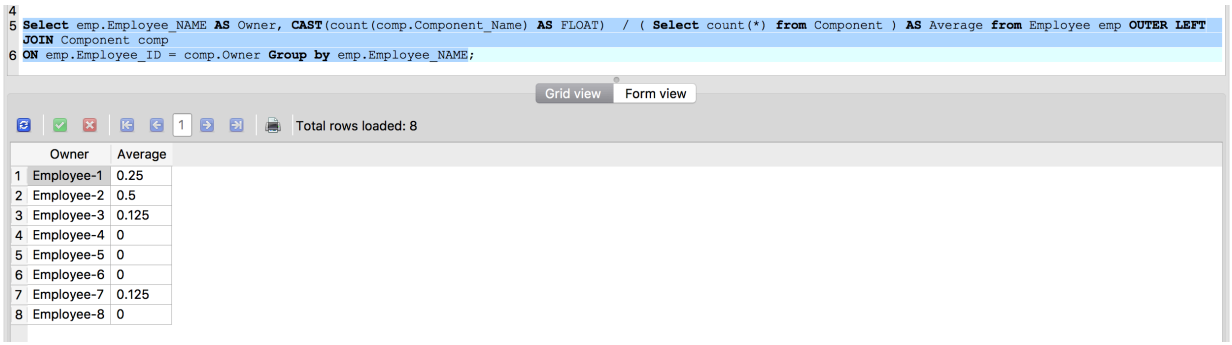
Grid view | Form view

Total rows loaded: 1

	Component Name	Component Versions
1	Chart generator	C11

4. What is the average number of components owned per person?

```
Select emp.Employee_NAME AS Owner, CAST(count(comp.Component_Name) AS FLOAT) /
( Select count(*) from Component ) AS Average from Employee emp OUTER LEFT JOIN
Component comp
ON emp.Employee_ID = comp.Owner Group by emp.Employee_NAME;
```



```
4
5 Select emp.Employee_NAME AS Owner, CAST(count(comp.Component_Name) AS FLOAT)  / ( Select count(*) from Component ) AS Average from Employee emp OUTER LEFT
  JOIN Component comp
6 ON emp.Employee_ID = comp.Owner Group by emp.Employee_NAME;
```

Grid view | Form view

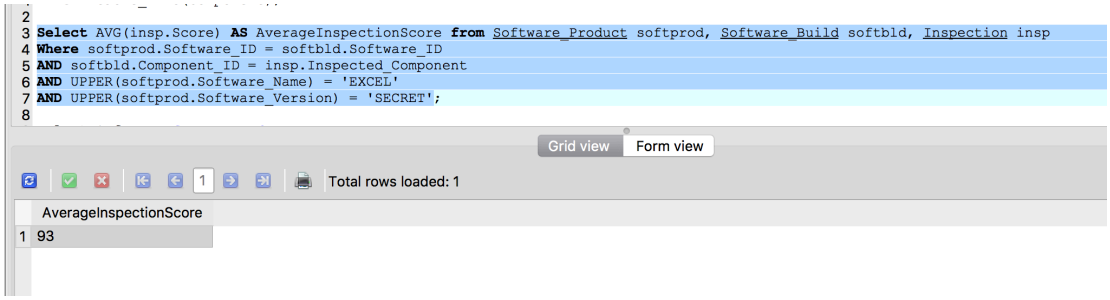
Total rows loaded: 8

	Owner	Average
1	Employee-1	0.25
2	Employee-2	0.5
3	Employee-3	0.125
4	Employee-4	0
5	Employee-5	0
6	Employee-6	0
7	Employee-7	0.125
8	Employee-8	0

Email: somyas@smu.edu
Student ID: 47304053 (Student ID)

5. What is the average score of all inspections for Excel secret?

```
Select AVG(insp.Score) AS AverageInspectionScore from Software_Product softprod,
Software_Build softbld, Inspection insp
Where softprod.Software_ID = softbld.Software_ID
AND softbld.Component_ID = insp.Inspected_Component
AND UPPER(softprod.Software_Name) = 'EXCEL'
AND UPPER(softprod.Software_Version) = 'SECRET';
```



6. List all employees by name, seniority, count of components assigned to them, count of inspections performed by them and their average inspection score.

```
Select
empvw.Employee_Name "Employee Name",
empvw.Seniority "Seniority",
IFNULL([ComponentOwned],0) AS NumberOfComponentOwned,
IFNULL([InspectedComponent],0) AS NumberofInspectionPerformed,
IFNULL([AverageInspectedScore],0) AS AverageInspectionScore
from Employee_Seniority_vw empvw
LEFT OUTER JOIN (Select Owner,Count(Component_Name) AS [ComponentOwned] From
Component group by Owner )
comp ON empvw.employee_ID = comp.Owner
LEFT OUTER JOIN (Select Own_BY, count(Inspection_date) AS
[InspectedComponent],avg(Score) AS [AverageInspectedScore] From Inspection group by
Own_BY ) insp ON empvw.employee_ID = insp.Own_BY;
```

```
Select
empvw.Employee_Name "Employee Name",
empvw.Seniority "Seniority",
IFNULL([ComponentOwned],0) AS NumberOfComponentOwned,
IFNULL([InspectedComponent],0) AS NumberofInspectionPerformed,
IFNULL([AverageInspectedScore],0) AS AverageInspectionScore
from Employee_Seniority_vw empvw
LEFT OUTER JOIN (Select Owner,Count(Component_Name) AS [ComponentOwned] From Component group by Owner )
comp ON empvw.employee_ID = comp.Owner
LEFT OUTER JOIN (Select Own_BY, count(Inspection_date) AS [InspectedComponent],avg(Score) AS [AverageInspectedScore] From Inspection
group by Own_BY ) insp ON empvw.employee_ID = insp.Own_BY;
```

Grid view | Form view

Total rows loaded: 8

	Employee Name	Seniority	NumberOfComponentOwned	NumberofInspectionPerformed	AverageInspectionScore
1	Employee-1	Senior	2	8	86
2	Employee-2	Senior	4	2	97.5
3	Employee-3	Senior	1	1	80
4	Employee-4	Senior	0	0	0
5	Employee-5	Junior	0	0	0
6	Employee-6	Junior	0	0	0
7	Employee-7	Junior	1	1	100
8	Employee-8	Newbie	0	0	0

7. Assume an inspection that results in a “ready” status costs $200, and all other inspections cost $100 each. How much did *OSF* in 2010 for inspections conducted by each seniority level?

The answer is based on Employee status as of 2010.

```
SELECT CASE WHEN ( (julianday(date(I.Inspection_date) ) - (julianday(Empl.Hire_date) ) ) < 365) THEN 'Newbie' WHEN ( (julianday(date(I.Inspection_date) ) - (julianday(Empl.Hire_date) ) ) > 1825) THEN 'Senior' WHEN (365 <= (julianday(date(I.Inspection_date) ) - (julianday(Empl.Hire_date) ) ) <= 1825) THEN 'Junior' END Seniority,
IFNULL((SUM(CASE WHEN I.Score > 90 THEN 200 WHEN I.Score <= 90 THEN 100 END)), 0) AS TotalCost
FROM Employee Empl
JOIN Inspection I ON Empl.Employee_ID = I.Own_BY and strftime('%Y', I.Inspection_date) = '2010'
GROUP BY Seniority,Empl.Employee_ID;
```

Query | History

```
SELECT CASE WHEN ( (julianday(date(I.Inspection_date) ) - (julianday(Empl.Hire_date) ) ) < 365) THEN 'Newbie' WHEN (
(julianday(date(I.Inspection_date) ) - (julianday(Empl.Hire_date) ) ) > 1825) THEN 'Senior' WHEN (365 <=
(julianday(date(I.Inspection_date) ) - (julianday(Empl.Hire_date) ) ) <= 1825) THEN 'Junior' END Seniority,
 IFNULL((SUM(CASE WHEN I.Score > 90 THEN 200 WHEN I.Score <= 90 THEN 100 END)), 0) AS TotalCost
FROM Employee Empl
JOIN Inspection I ON Empl.Employee_ID = I.Own_BY and strftime('%Y', I.Inspection_date) = '2010'
GROUP BY Seniority,Empl.Employee_ID;
```

Grid view | Form view

Total rows loaded: 1

	Seniority	TotalCost
1	Senior	800

To test I created one more inspection as shown below

	Inspected_Component	Inspection_date	Own_BY	Score	Description
1	1	2010-02-14	10100	100	legacy code which is already approved
2	3	2010-02-22	10100	55	too many hard coded parameters, the software must be r
3	3	2010-02-24	10100	78	improved, but only handles DB2 format
4	9	2010-02-24	10400	78	This is for Testing purposes
5	3	2010-02-26	10100	95	2017-01-01
6	3	2010-02-28	10100	100	satisifed

Employee 10400 Hire Date is '2008-11-01' which makes him junior at the time of Inspection.

Query History

```
SELECT CASE WHEN ( (julianday(date(I.Inspection_date) ) - (julianday(Empl.Hire_date) ) ) < 365) THEN 'Newbie' WHEN (
(julianday(date(I.Inspection_date) ) - (julianday(Empl.Hire_date) ) ) > 1825) THEN 'Senior' WHEN (365 <=
(julianday(date(I.Inspection_date) ) - (julianday(Empl.Hire_date) ) ) <= 1825) THEN 'Junior' END Seniority,
 IFNULL((SUM(CASE WHEN I.Score > 90 THEN 200 WHEN I.Score <= 90 THEN 100 END)), 0) AS TotalCost
FROM Employee Empl
JOIN Inspection I ON Empl.Employee_ID = I.Own_BY and strftime('%Y', I.Inspection_date) = '2010'
GROUP BY Seniority,Empl.Employee_ID;
```

Grid view Form view

Total rows loaded: 2

	Seniority	TotalCost
1	Junior	100
2	Senior	800

8. Demonstrate the adding of a new inspection by employee 10400 on Pen driver - P01 held on 8/15/2017 with the score of 60 and description of "needs rework, introduced new errors".

Insert into Inspection ('Inspected_Component','Inspection_Date','Own_By','Score','Description')
Values (6,'2017-08-15',10400,60,'needs rework, introduced new errors');

Select * from Inspection where Inspected_Component = 6 and Own_By = 10400;

```
Insert into Inspection ('Inspected_Component','Inspection_Date','Own_By','Score','Description')
Values (6,'2017-08-15',10400,60,'needs rework, introduced new errors');

Select * from Inspection where Inspected_Component = 6 and Own_By = 10400;
```

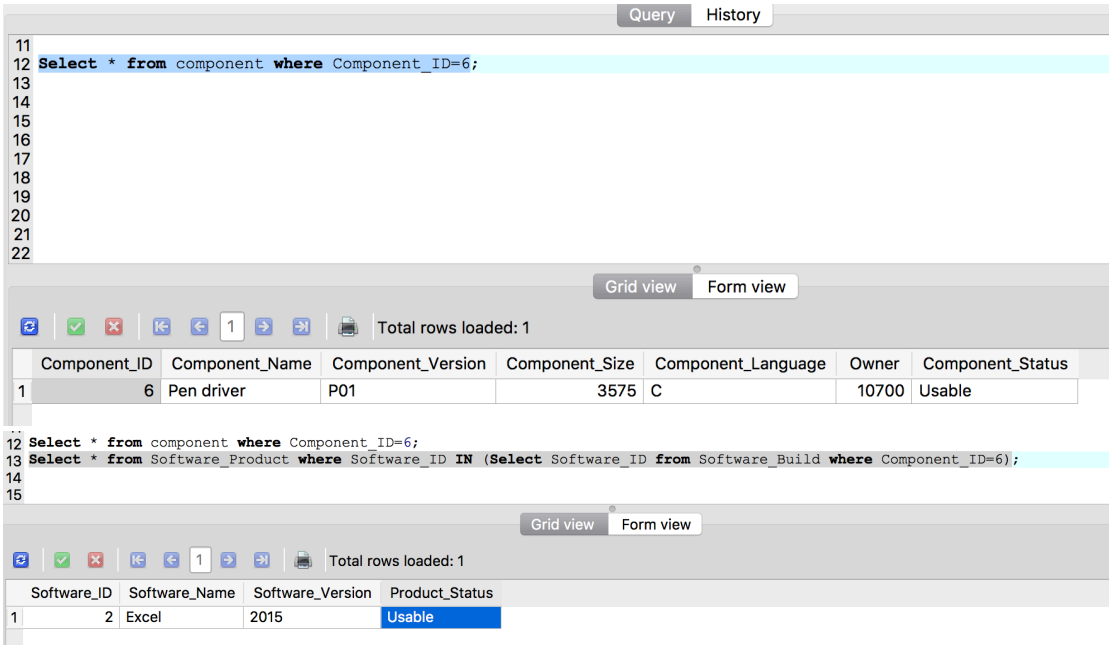
Grid view Form view

Total rows loaded: 1

	Inspected_Component	Inspection_date	Own_BY	Score	Description
1	6	2017-08-15	10400	60	needs rework, introduced new errors

Email: somyas@smu.edu
Student ID: 47304053 (Student ID)

Before Insert



```
Select * from component where Component_ID=6;
```

Query | History

Grid view | Form view

Total rows loaded: 1

	Component_ID	Component_Name	Component_Version	Component_Size	Component_Language	Owner	Component_Status
1	6	Pen driver	P01	3575	C	10700	Usable

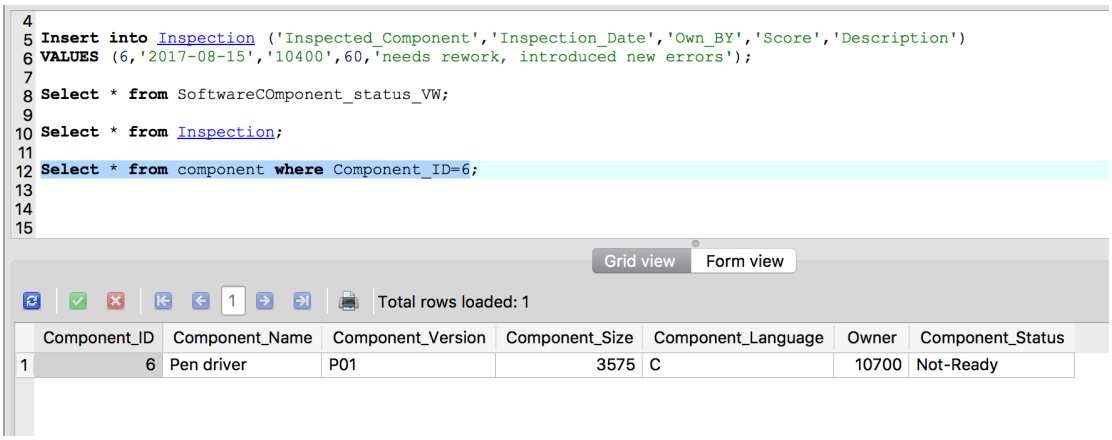
```
Select * from component where Component_ID=6;
Select * from Software_Product where Software_ID IN (Select Software_ID from Software_Build where Component_ID=6);
```

Grid view | Form view

Total rows loaded: 1

	Software_ID	Software_Name	Software_Version	Product_Status
1	2	Excel	2015	Usable

After Insert Status changes to Not-Ready

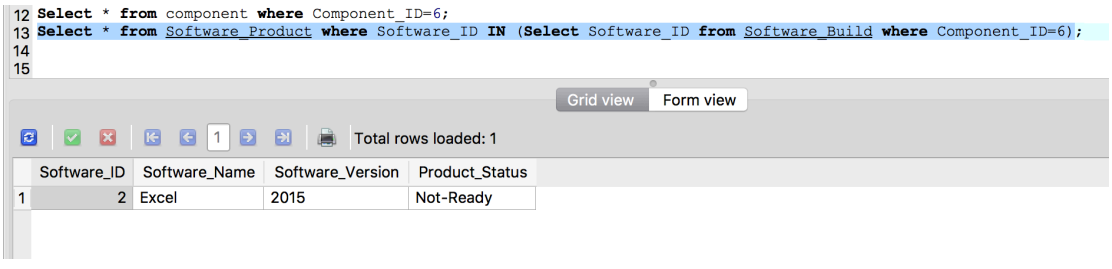


```
Insert into Inspection ('Inspected_Component','Inspection_Date','Own_BY','Score','Description')
VALUES (6,'2017-08-15','10400',60,'needs rework, introduced new errors');

Select * from SoftwareCOmponent_status_VW;

Select * from Inspection;

Select * from component where Component_ID=6;
```

Grid view | Form view

Total rows loaded: 1

	Component_ID	Component_Name	Component_Version	Component_Size	Component_Language	Owner	Component_Status
1	6	Pen driver	P01	3575	C	10700	Not-Ready



```
Select * from component where Component_ID=6;
Select * from Software_Product where Software_ID IN (Select Software_ID from Software_Build where Component_ID=6);
```

Grid view | Form view

Total rows loaded: 1

	Software_ID	Software_Name	Software_Version	Product_Status
1	2	Excel	2015	Not-Ready

Email: somyas@smu.edu
Student ID: 47304053 (Student ID)

9. A) Demonstrate adding a new component to Excel 2018beta. This new component is named "Dynamic Table Interface", version D01, and was written in JavaScript by person 10400, size = 775.

First, I inserted Programming Language entry for JavaScript since Foreign key was being violated in Component table.
Then I inserted the entry into Component.

```
3
4 Select * from Component where Component_Name = 'Dynamic Table Interface';
```

Grid view | Form view

Total rows loaded: 1

	Component_ID	Component_Name	Component_Version	Component_Size	Component_Language	Owner	Component_Status
1	9	Dynamic Table Interface	D01	775	Javascript	10400	Not-Ready

Query | History

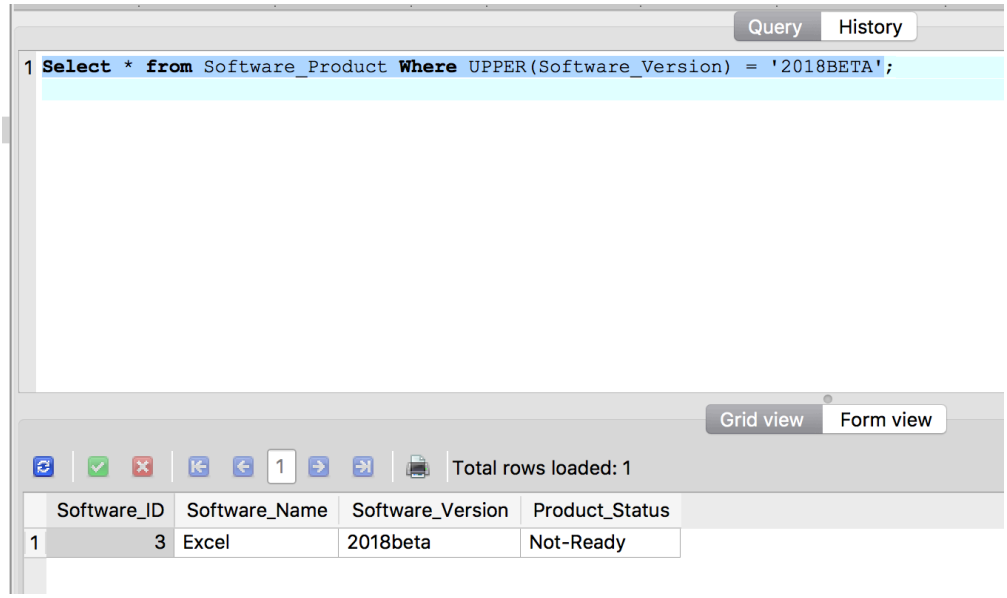
```
1 Select softprod.Software_Name,softprod.Software_Version,comp.Component_Name from Software_Product softprod, Software_Build softbuild,
  Component comp
2 Where softprod.Software_ID = softbuild.Software_ID
3 AND comp.Component_ID = softbuild.Component_ID
4 AND softprod.Software_Name = 'Excel'
5 AND softprod.Software_Version = '2018beta'
6
```

Grid view | Form view

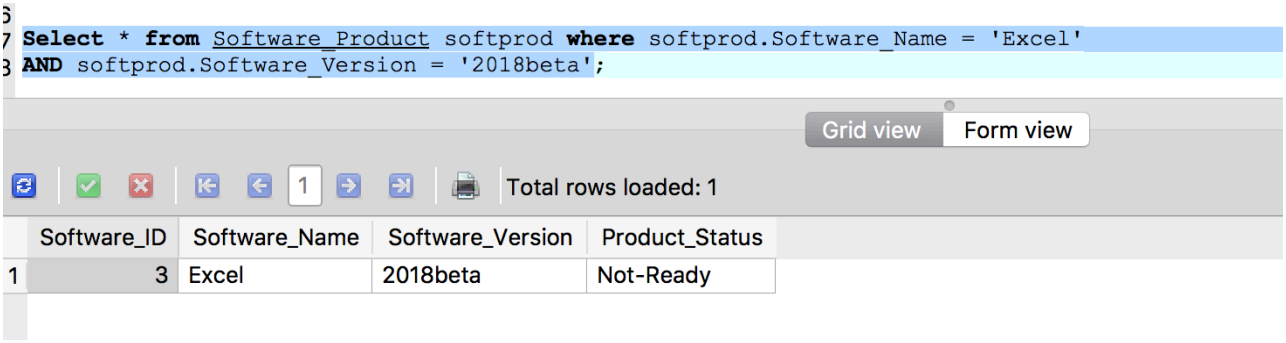
Total rows loaded: 4

	Software_Name	Software_Version	Component_Name
1	Excel	2018beta	Chart generator
2	Excel	2018beta	Dynamic Table Interface
3	Excel	2018beta	Keyboard Driver
4	Excel	2018beta	Touch Screen Driver

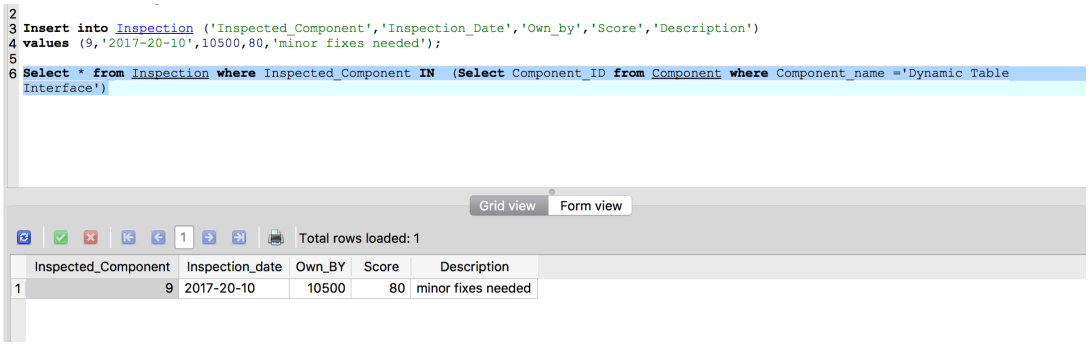
B) What is the Excel 2018beta product status?
Original Status (before adding then New Component):



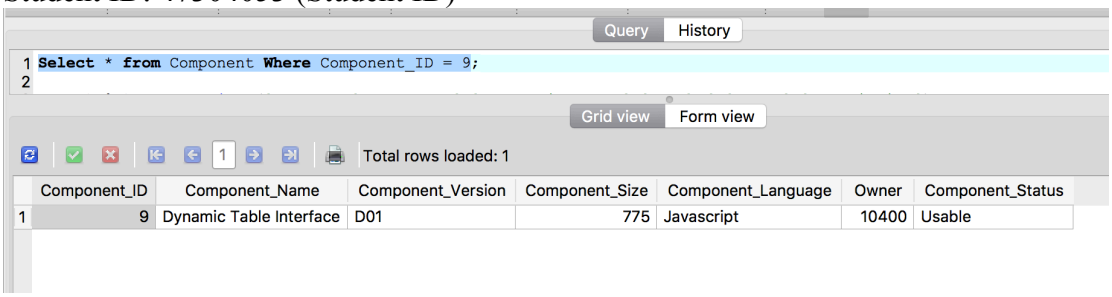
New Status:



10. A) Demonstrate the adding of an inspection on the component you just added. This inspection occurred on 11/20/2017 by inspector 10500, with a score of 80, and note of "minor fixes needed".



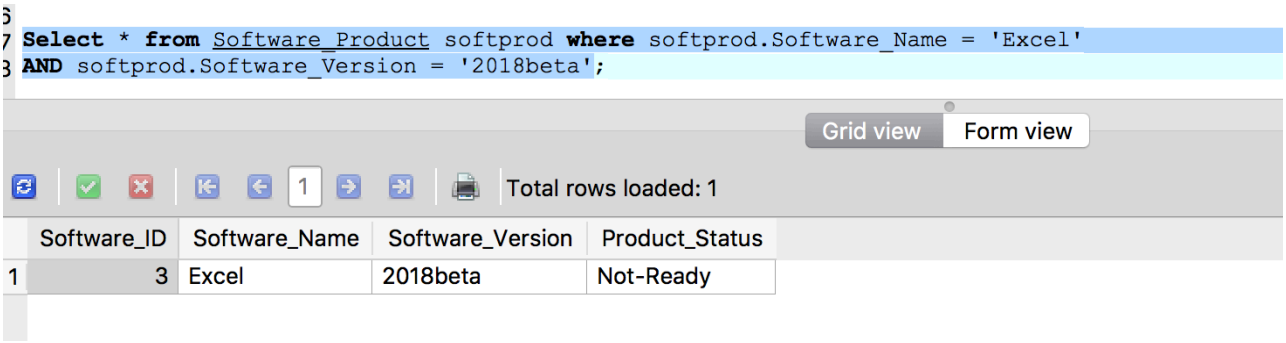
Email: somyas@smu.edu
Student ID: 47304053 (Student ID)



	Component_ID	Component_Name	Component_Version	Component_Size	Component_Language	Owner	Component_Status
1	9	Dynamic Table Interface	D01	775	Javascript	10400	Usable

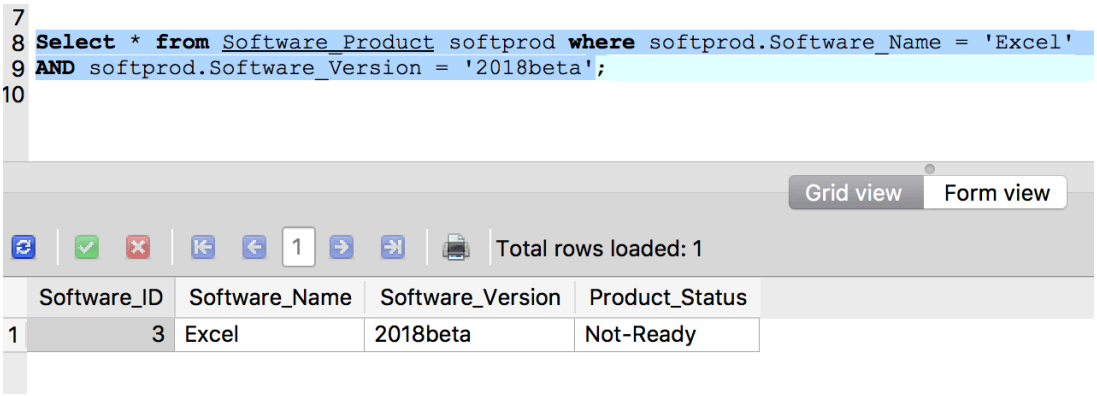
B) What is the Excel 2018beta product status?

Old Status:



	Software_ID	Software_Name	Software_Version	Product_Status
1	3	Excel	2018beta	Not-Ready

New Status



	Software_ID	Software_Name	Software_Version	Product_Status
1	3	Excel	2018beta	Not-Ready

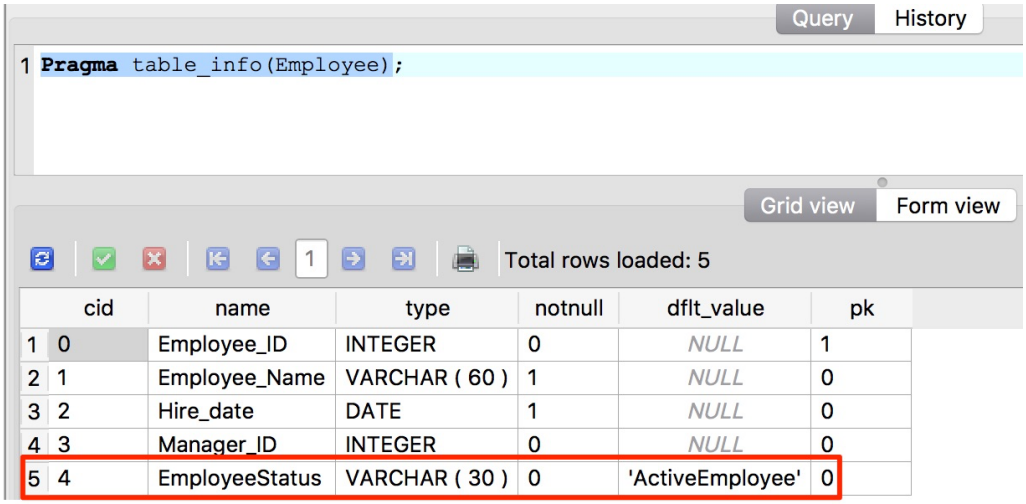
GRADUATE:

11. Person 10700 has decided to leave *OSF* for other employment. Implement a solution for this situation.

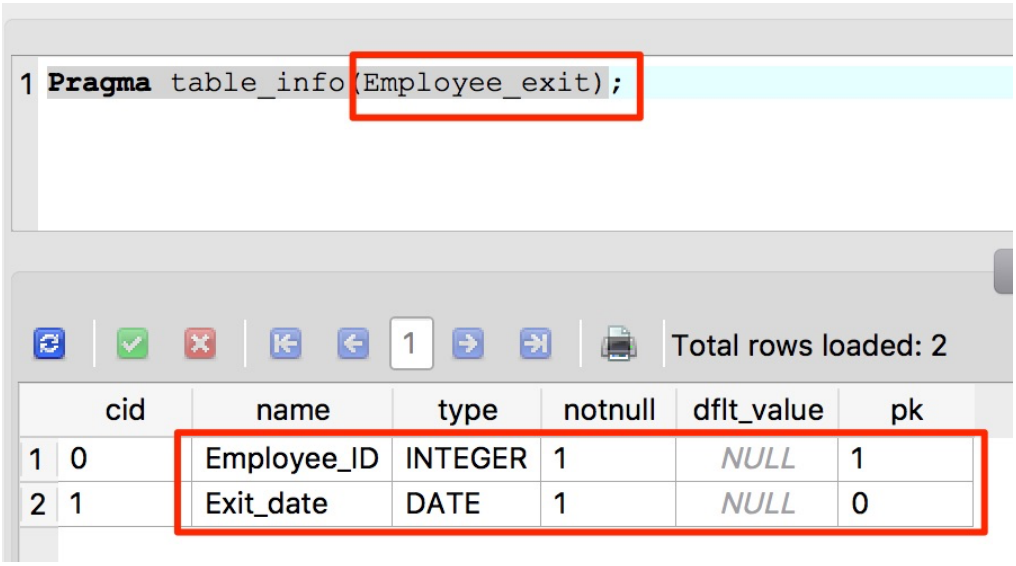
To implement the solution for this I decided not to delete the Employee details from "Employee" table but update the Employee record so that it can reflect active vs an ex-employee. In order to do this, I decided to make following structural changes to my schema;

1. Added column EmployeeStatus which was char (30) with only two possible values "ActiveEmployee" vs "ExEmployee". **By Default, it will be "ActiveEmployee".**

2. Created a table "Employee_exit" which have two columns Employee_ID and Exit_Date which is Date the employee left the company.



	cid	name	type	notnull	dflt_value	pk
1	0	Employee_ID	INTEGER	0	NULL	1
2	1	Employee_Name	VARCHAR (60)	1	NULL	0
3	2	Hire_date	DATE	1	NULL	0
4	3	Manager_ID	INTEGER	0	NULL	0
5	4	EmployeeStatus	VARCHAR (30)	0	'ActiveEmployee'	0



	cid	name	type	notnull	dflt_value	pk
1	0	Employee_ID	INTEGER	1	NULL	1
2	1	Exit_date	DATE	1	NULL	0

Following Conditions were implemented using triggers;

1. If an Employee Status is updated to "ExEmployee" with no date defined for Exit date, system will Insert a record into Employee_exit and default the End date as Sysdate.
2. Checks were in place to ensure Exit Date cannot be Future date.
3. Check were in place to ensure Employee Status can only be "Active Employee" or "ExEmployee".
4. While inserting or updating a record in Component Table the check was in place to ensure Owner is not an ExEmployee.
5. While inserting or updating a record in Inspection Table the check was in place to ensure Owner is not an ExEmployee.
6. While inserting or updating Employee record the Manager check for Active employee was created.

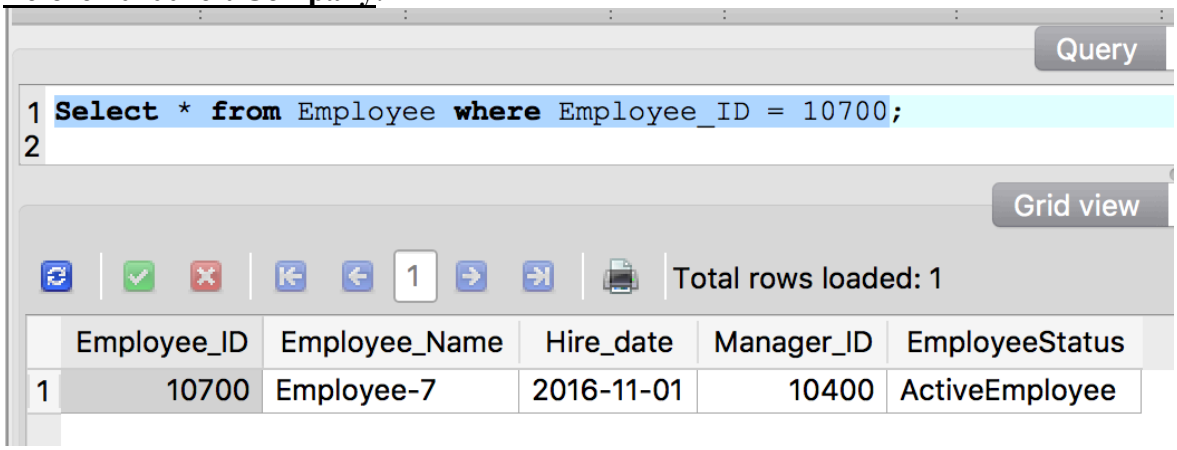
7. Only the Active Employee Seniority is calculated.

To implement the Employee leaving the OSF below triggers were created;

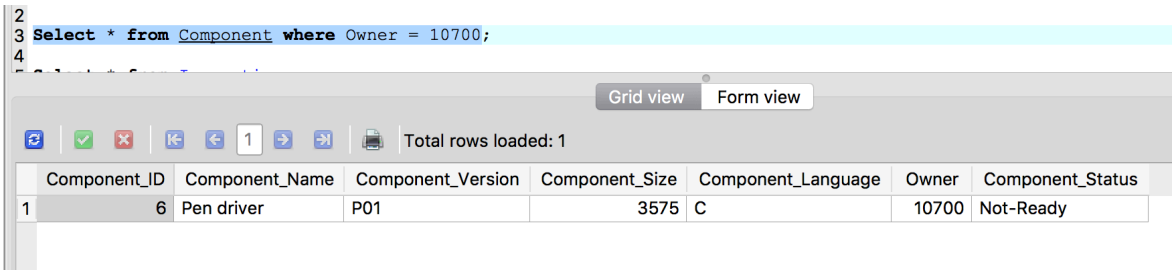
1. If the Employee has no manager
a. All the Employee who had their manager as leaving employee then their Manager will be updated to senior most employee of the company.
b. All the component owned by Employee will be updated to have Senior most employee as its owner.
c. No update to Inspection table will be done.
2. If the Employee has a manager
a. All the Employee who had their manager as leaving employee then their Manager will be updated to his manager.
b. All the component owned by Employee will be updated to have his Manager as its owner.
c. No update to Inspection table will be done.

Now When Employee "10700" leaves Company below changes happened;

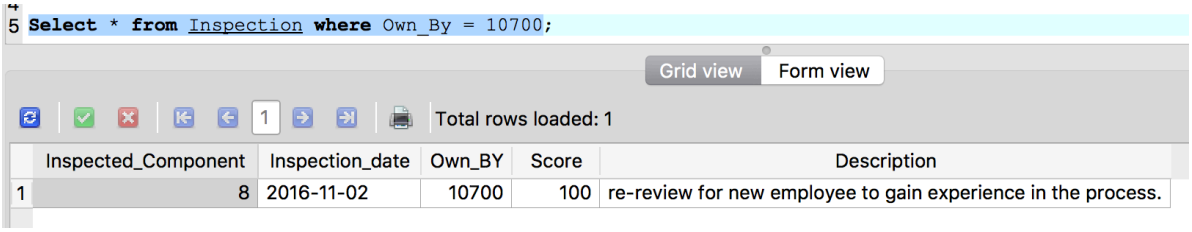
Before 10700 left Company:



	Employee_ID	Employee_Name	Hire_date	Manager_ID	EmployeeStatus
1	10700	Employee-7	2016-11-01	10400	ActiveEmployee



	Component_ID	Component_Name	Component_Version	Component_Size	Component_Language	Owner	Component_Status
1	6	Pen driver	P01	3575	C	10700	Not-Ready



	Inspected_Component	Inspection_date	Own_BY	Score	Description
1	8	2016-11-02	10700	100	re-review for new employee to gain experience in the process.

```
6
7 Select * from Employee_exit where Employee_ID = 10700;
```

Total rows loaded: 0

Employee_ID	Exit_date

```
6
7 Select * from Employee_Seniority_vw where Employee_ID = 10700;
```

Grid view

Total rows loaded: 1

	Employee_ID	Employee_Name	Seniority
1	10700	Employee-7	Junior

After 10700 left Company:
The status of Employee is updated to "ExEmployee"

```
2
3 Select * from Employee where Employee_ID = 10700;
4
```

Grid view

Total rows loaded: 1

	Employee_ID	Employee_Name	Hire_date	Manager_ID	EmployeeStatus
1	10700	Employee-7	2016-11-01	10400	ExEmployee

Email: somyas@smu.edu
Student ID: 47304053 (Student ID)

```
Select * from Employee_exit where Employee_ID = 10700;
```

Grid v

Total rows loaded: 1

	Employee_ID	Exit_date
1	10700	2017-12-05

The Component Owned by "10700" is now updated to his manager "10400"

```
Select * from Component where Component_ID = 6;
```

Grid view | Form view

Total rows loaded: 1

	Component_ID	Component_Name	Component_Version	Component_Size	Component_Language	Owner	Component_Status
1	6	Pen driver	P01	3575	C	10400	Not-Ready

No Changes to Inspection:

```
Select * from Inspection where Own_By = 10700;
```

Grid view | Form view

Total rows loaded: 1

	Inspected_Component	Inspection_date	Own_BY	Score	Description
1	8	2016-11-02	10700	100	re-review for new employee to gain experience in the process.

Record from Employee Seniority is removed.

```
Select * from Employee_Seniority_vw where Employee_ID = 10700;
```

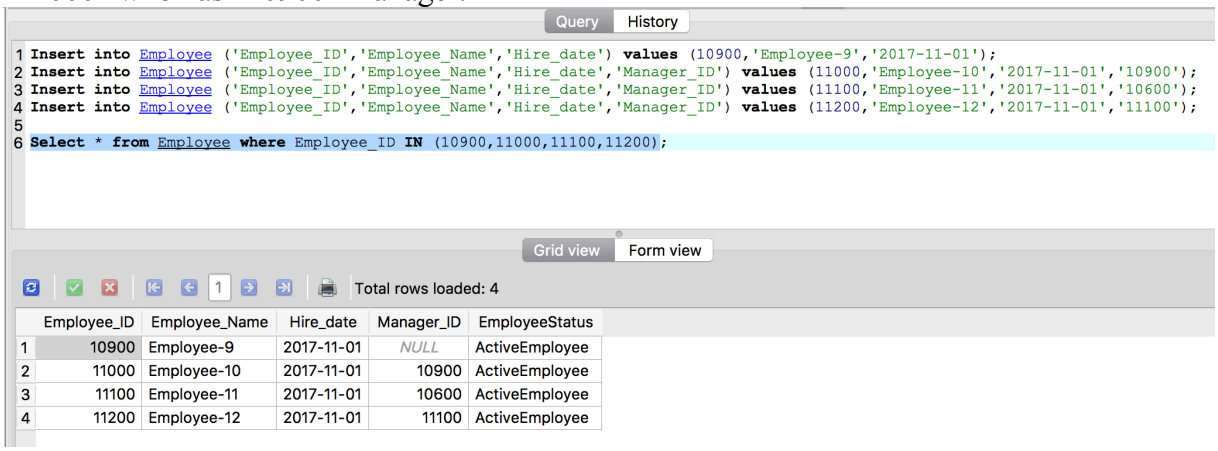
Grid view

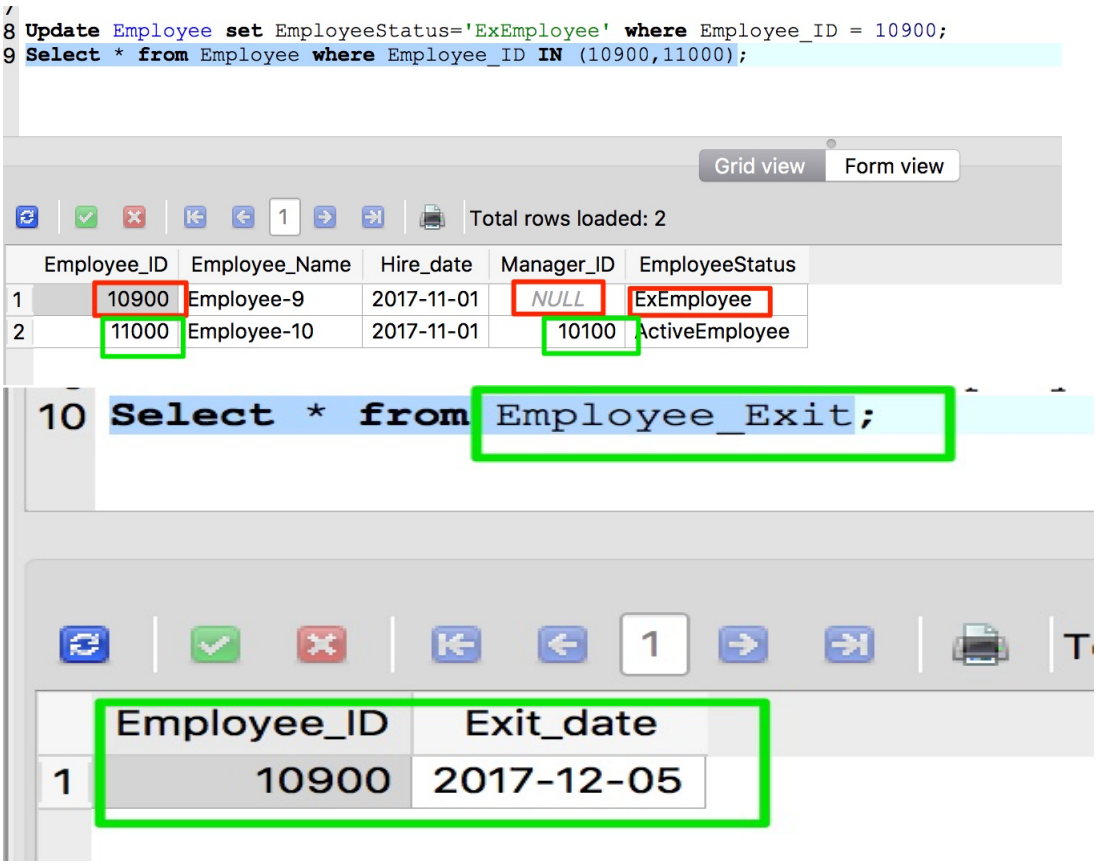
Total rows loaded: 0

Employee_ID	Employee_Name	Seniority

Email: somyas@smu.edu
Student ID: 47304053 (Student ID)

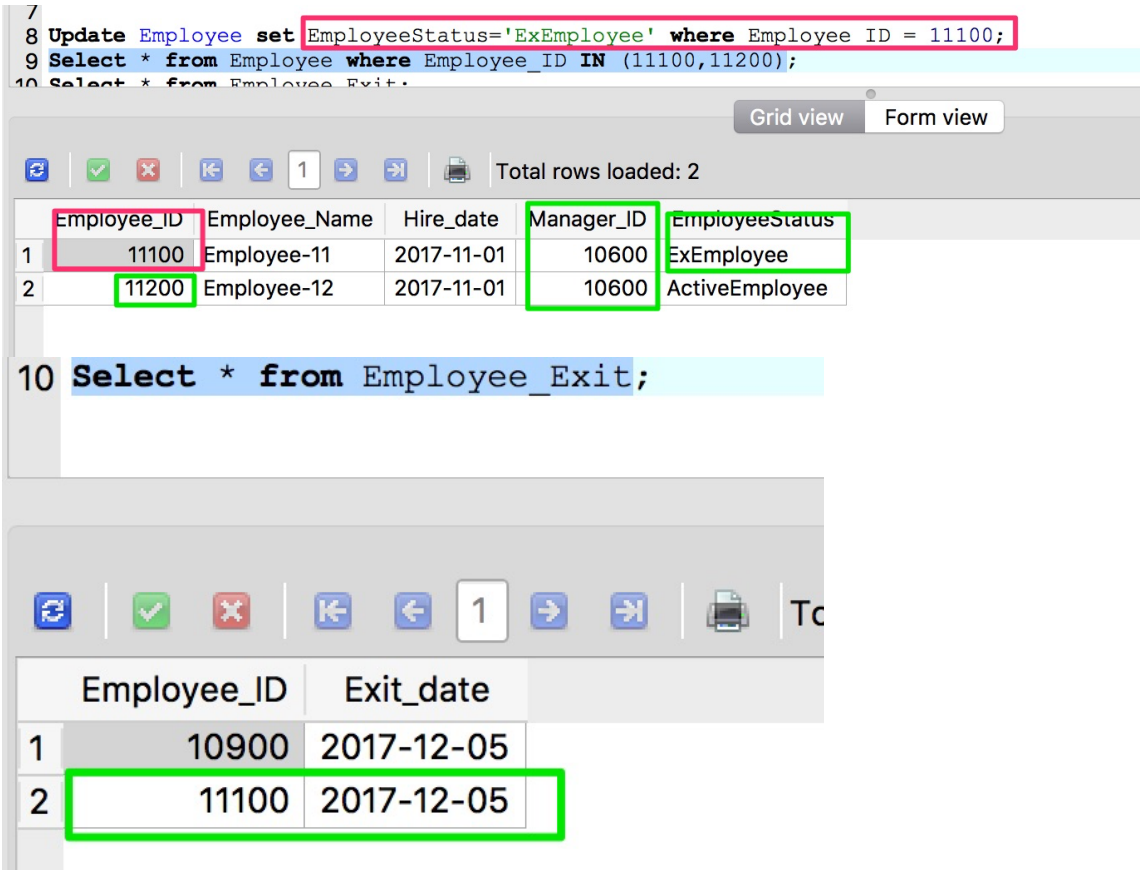
Employee with No Manager is leaving;
Assuming we have Employee "10900" and he has no manager. There is another employee "11000" who has "10900" manager.





So, after 10900 left the factory, "11000" manager was updated to "10100" the senior most employee and an Exit date for Employee was assigned.

Employee with Manager is leaving;



So once "111000" left, manager for "11200" was updated to manager of "11100" which was "10600".

Email: somyas@smu.edu
Student ID: 47304053 (Student ID)

Final Database details

```
1 Select type,count(*) from sqlite_master group by type;
```

Total rows loaded: 4

	type	count(*)
1	index	7
2	table	8
3	trigger	46
4	view	3

Query History

```
1 Select tbl_name,type,count(*) from sqlite_master group by tbl_name,type Order by tbl_name;
```

Grid view Form view

Total rows loaded: 22

	tbl_name	type	count(*)
1	Component	index	1
2	Component	table	1
3	Component	trigger	5
4	Employee	index	1
5	Employee	table	1
6	Employee	trigger	19
7	Employee_Seniority_vw	view	1
8	Employee_exit	table	1
9	Employee_exit	trigger	5
10	Inspection	index	1
11	Inspection	table	1
12	Inspection	trigger	14
13	Programming_language	index	2
14	Programming_language	table	1
15	SoftwareComponent_status_vw	view	1
16	SoftwareProduct_status_vw	view	1
17	Software_Build	index	1
18	Software_Build	table	1
19	Software_Product	index	1
20	Software_Product	table	1
21	Software_Product	trigger	3
22	sqlite_sequence	table	1