

CSE 5330/7330 Fall 2017
Phase 3 Functional Requirements
DHAVAL GOGRI
47444609

Using your database populated with the data provided.

Everyone: Write Queries (and show the results) to answer the following questions:

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

`select * from software_products;`

The screenshot shows a database management tool interface. The top pane displays a SQL query for 'Query 1' in the 'fodm_project' database. The query is as follows:

```
-- Project Phase 3 --  
-- select * from inspection;  
-- select * from components_in_software_products;  
-- select * from components;  
-- select * from employees;  
-- select * from software_products;  
-- select * from programming_language;  
  
-- 1  
* software_products;  
  
-- 2  
select Employees.name, Components.component name, Components.version
```

The bottom pane shows the 'Result Grid' with the following data:

| name | version | software_status |
|-------|----------|-----------------|
| Excel | 2010 | Readv |
| Excel | 2015 | usable |
| Excel | 2018beta | not-readv |
| Excel | secret | not-readv |
| NULL | NULL | NULL |

The bottom pane also shows the 'Output' section with the following action output:

| # | Time | Action | Message |
|-----|----------|-----------------------------------------------------------------------------------------------|-------------------|
| 273 | 15:23:33 | insert into inspection(inspection_id, component_name, version, inspection_date, by_who, sc... | 1 row(s) affected |
| 274 | 15:23:45 | select * from software_products LIMIT 0, 10000 | 4 row(s) returned |

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

```
select Employees.name,Components.component_name,Components.version
from Employees, Components
where components.component_status like 'not-ready' and components.comp_owner =
employees.id;
```

The screenshot shows a database query editor window titled "Query 1" with a tab labeled "fodm_project". The query text is as follows:

```
-- select * from components;
-- select * from employees;
-- select * from software_products;
-- select * from programming_language;

-- 1
select * from software_products;

-- 2
Employees.name,Components.component_name,Components.version
Employees, Components
components.component_status like 'not-ready' and components.comp_owner = employees.id;

-- 3
```

Below the query editor, the "Result Grid" is displayed with the following data:

| | name | component_name | version |
|------------|-----------------|----------------|---------|
| Employee-2 | Chart generator | C11 | |

At the bottom, the "Output" pane shows the execution log:

| # | Time | Action | Message |
|-----|----------|--------------------------------------------------------------------------------------|-------------------|
| 274 | 15:23:45 | select * from software_products LIMIT 0, 10000 | 4 row(s) returned |
| 275 | 15:24:53 | select Employees.name,Components.component_name,Components.version from Employees... | 1 row(s) returned |

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

```
select Components.component_name,Components.version
from components
left outer join inspection on Components.component_name = inspection.component_name and
components.version = inspection.version
where inspection.component_name is null;
```

The screenshot shows a database query editor interface. The top toolbar includes icons for file operations, query execution, and a 'Limit to 10000 rows' dropdown. The main text area contains SQL code with line numbers 539 to 555. The code includes comments and two SQL queries. The first query selects all columns from 'software_products'. The second query selects 'Employees.name', 'Components.component_name', and 'Components.version' from 'Employees' and 'Components' tables, filtered by 'components.component_status like 'not-ready'' and 'components.comp_owner = employees.id'. The third query is a left outer join between 'Components' and 'inspection' tables on 'component_name' and 'version'.

Below the code editor, there is a 'Result Grid' section with a table showing the results of the query. The table has two columns: 'component_name' and 'version'. The first row shows 'Chart generator' and 'C11'.

At the bottom, there is an 'Output' section with a table showing the execution results. The table has four columns: '#', 'Time', 'Action', and 'Message'. The first row shows a successful execution of the first query at 15:24:53, returning 1 row(s). The second row shows a successful execution of the second query at 15:25:16, returning 1 row(s).

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

```
select avg(tableTemp.componentsCount)
from ( select count(components.comp_id) as componentsCount
from components
group by components.comp_owner) as tableTemp;
```

The screenshot shows a database query editor window with a query that has been executed. The query is as follows:

```
545 • select Employees.name,Components.component_name,Components.version
546 from Employees, Components
547 where components.component_status like 'not-ready' and components.comp_owner = employees.id;
548
549 -- 3
550 • select Components.component_name,Components.version
551 from components
552 left outer join inspection on Components.component_name = inspection.component_name and components.version = in
553 where inspection.component_name is null;
554
555 -- 4
556 • (tableTemp.componentsCount)
557 ( (components.comp_id) componentsCount
558 components
559 components.comp_owner) tableTemp;
560
561 -- 5
```

The result grid shows the following data:

| avg(tableTemp.componentsCount) |
|--------------------------------|
| 2.0000 |

The bottom section of the screenshot shows the 'Action Output' window with the following log:

| # | Time | Action | Message |
|-----|----------|------------------------------------------------------------------------------------------|-------------------|
| 276 | 15:25:16 | select Components.component_name,Components.version from components left outer join i... | 1 row(s) returned |
| 277 | 15:25:38 | select avg(tableTemp.componentsCount) from (select count(components.comp_id) as comp... | 1 row(s) returned |

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

```
select avg(inspection.score)
from inspection
where (component_name,version) in (select component_name,version
                                   from components
                                   where comp_id in (select comp_id
                                                    from
components_in_software_products
                                                    where
components_in_software_products.name like 'Excel' and
components_in_software_products.version like 'secret'));
```

The screenshot shows a database query editor window titled "Query 1" with a tab labeled "fodm_project". The query is as follows:

```
-- 4
select avg(tableTemp.componentsCount)
from ( select count(components.comp_id) as componentsCount
      from components
      group by components.comp_owner) as tableTemp;
-- 5
      (inspection.score)
      inspection
      (component_name,version) (      component_name,version
      components
      comp_id (      comp_id
      components_in_software_products
      components_in_software_products.      'Excel' components_in_s
-- 6
```

The query is executed, and the results are displayed in a "Result Grid" below the query editor. The grid shows a single row with the value 93.0000 for the expression avg(inspection.score).

| avg(inspection.score) |
|-----------------------|
| 93.0000 |

Below the result grid, there is an "Output" section with a table showing the execution of the query:

| # | Time | Action | Message |
|-----|----------|-----------------------------------------------------------------------------------------------|-------------------|
| 277 | 15:25:38 | select avg(tableTemp.componentsCount) from (select count(components.comp_id) as comp... | 1 row(s) returned |
| 278 | 15:25:58 | select avg(inspection.score) from inspection where (component_name,version) in (select com... | 1 row(s) returned |

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 e.name, e.seniority, count(distinct c.comp_id) as Assigned_Components, count(distinct
i.inspection_id)
as Inspection_Performed, avg(i.score) as Average_Score
from components c right join employees e on e.id = c.comp_owner
left join inspection i on e.id = i.by_who
group by e.id;
```

The screenshot shows a database query tool interface. At the top, there's a tab labeled 'Query 1' and 'fodm_project'. Below the toolbar, a 'Find' bar is visible. The main area displays a SQL query with line numbers 552 to 563. The query is as follows:

```
-- 6
552
553
554 e. , e.seniority, ( c.comp_id) Assigned_Components, ( i.inspection_id)
555 Inspection_Performed, (i.score) Average_Score
556 components c employees e e.id = c.comp_owner
557 inspection i e.id = i.by_who
558 e.id;
559
560
561
562
563
```

Below the query editor, the 'Result Grid' is shown. It has columns: name, seniority, Assigned_Components, Inspection_Performed, and Average_Score. The data is as follows:

| name | seniority | Assigned_Components | Inspection_Performed | Average_Score |
|------------|-----------|---------------------|----------------------|---------------|
| Employee-1 | senior | 2 | 8 | 86.0000 |
| Employee-2 | senior | 4 | 2 | 97.5000 |
| Employee-3 | senior | 1 | 1 | 80.0000 |
| Employee-4 | senior | 0 | 0 | NULL |
| Employee-5 | junior | 0 | 0 | NULL |
| Employee-6 | junior | 0 | 0 | NULL |
| Employee-7 | junior | 1 | 1 | 100.0000 |
| Employee-8 | newbie | 0 | 0 | NULL |

At the bottom, the 'Output' section shows two messages:

- 749 16:55:44 select e.name, e.seniority, count(distinct c.comp_id) as Components_Assigned, count(distinct... 8 row(s) returned
- 750 16:57:40 select e.name, e.seniority, count(distinct c.comp_id) as Assigned_Components, count(distinct... 8 row(s) returned

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?

```
select employees.seniority ,
CASE status
  when 'ready' then count(status)*200
  when 'not-ready' then count(status)*100
  when 'usable' then count(status)*100
end as cost
from employees
join inspection on employees.id = inspection.by_who
where DATE_FORMAT( inspection.inspection_date, "%Y" ) = "2010"
group by employees.seniority;
```

The screenshot shows a database query editor window titled "Query 1" with a tab labeled "fodm_project". The query is as follows:

```
-- 7
select employees.seniority ,
CASE status
  when 'ready' then count(status)*200
  when 'not-ready' then count(status)*100
  when 'usable' then count(status)*100
end as cost
from employees
join inspection on employees.id = inspection.by_who
where DATE_FORMAT( inspection.inspection_date, "%Y" ) = "2010"
group by employees.seniority;
```

The results are displayed in a "Result Grid" with two columns: "seniority" and "cost". The data shows that for the "senior" seniority level, the cost is 1000.

| seniority | cost |
|-----------|------|
| senior | 1000 |

Below the result grid, the "Output" section shows the "Action Output" for the query. It indicates that the query was executed successfully and returned 1 row(s).

| # | Time | Action | Message |
|-----|----------|------------------------------------------------------------------------------------------------|-------------------|
| 278 | 15:25:58 | select avg(inspection.score) from inspection where (component_name.version) in (select com... | 1 row(s) returned |
| 279 | 15:26:25 | select employees.seniority , CASE status when 'ready' then count(status)*200 when 'not-read... | 1 row(s) returned |

Everyone: Demonstrate = show the SQL command(s) and result

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(inspection_id, component_name, version, inspection_date, by_who, score, description) values(13, "Pen Driver", "P01", 08/15/2017, 10400, 60, "needs rework, introduced new errors");

The screenshot shows a database management tool interface. At the top, there's a toolbar with various icons and a search bar. Below the search bar, a SQL query is entered in a text area:

```
-- 10 A --  
-- Demonstrate the adding of an inspection on the component you just added. This inspection occurred on 11/20/2  
insert into inspection(inspection_id, component_name, version, inspection_date, by_who, score, description) val
```

Below the query editor, there's a "Result Grid" showing a table of inspection records. The table has columns: inspection_id, component_name, version, inspection_date, by_who, score, description, and status. The records are numbered 1 through 13, with the last record (13) being the one just inserted.

| inspection_id | component_name | version | inspection_date | by_who | score | description | status |
|---------------|---------------------|---------|---------------------|--------|-------|---------------------------------------------------|-----------|
| 1 | Keyboard Driver | K11 | 2010-02-14 00:00:00 | 10100 | 100 | legacy code which is already approved | Ready |
| 2 | Touch Screen Driver | T00 | 2017-06-01 00:00:00 | 10200 | 95 | initial release ready for usage | Ready |
| 3 | Dbase Interface | D00 | 2010-02-22 00:00:00 | 10100 | 55 | too many hard coded parameters. the software... | not-ready |
| 4 | Dbase Interface | D00 | 2010-02-24 00:00:00 | 10100 | 78 | improved. but only handles DB2 format | usable |
| 5 | Dbase Interface | D00 | 2010-02-26 00:00:00 | 10100 | 95 | Okav. handles DB3 format. | Ready |
| 6 | Dbase Interface | D00 | 2010-02-28 00:00:00 | 10100 | 100 | satisfied | Ready |
| 7 | Dbase Interface | D01 | 2011-05-01 00:00:00 | 10200 | 100 | Okav ready for use | Ready |
| 8 | Pen Driver | P01 | 2017-07-15 00:00:00 | 10300 | 80 | Okav ready for beta testing | usable |
| 9 | Math unit | A01 | 2014-06-10 00:00:00 | 10100 | 90 | almost ready | usable |
| 10 | Math unit | A02 | 2014-06-15 00:00:00 | 10100 | 70 | Accuracy problems! | not-ready |
| 11 | Math unit | A02 | 2014-06-30 00:00:00 | 10100 | 100 | Okav problems fixed | Ready |
| 12 | Math unit | A02 | 2016-11-02 00:00:00 | 10700 | 100 | re-review for new employee to gain experience ... | Ready |
| 13 | Pen Driver | P01 | 0000-00-00 00:00:00 | 10400 | 60 | needs rework. introduced new errors | not-ready |

Below the result grid, there's an "Output" section showing the execution of the query. It includes a table with columns: #, Time, Action, and Message. The output shows that the insert statement was executed successfully, affecting 1 row(s).

| # | Time | Action | Message |
|-----|----------|-----------------------------------------------------------------------------------------------|--------------------|
| 280 | 15:27:20 | insert into inspection(inspection_id, component_name, version, inspection_date, by_who, sc... | 1 row(s) affected |
| 281 | 15:27:32 | select * from inspection LIMIT 0, 10000 | 13 row(s) returned |

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.

1. insert into programming_language values('JavaScript','Current');

2. insert into Components(comp_id, component_name, version, component_size, prog_language, comp_owner) values(9, 'Dynamic Table Interface', 'D01', 775, 'JavaScript', 10400);

The screenshot displays a database management interface. At the top, a 'Query 1' tab is active, showing a SQL script with two delete statements. Below the script, a 'Result Grid' tab shows a table with 9 rows and 7 columns. The table contains data for various components, including 'Dynamic Table Interface' (row 9). At the bottom, an 'Output' panel shows two successful actions: an insert statement affecting 1 row and a select statement returning 9 rows.

Query 1 fodm_project x

Find Find Manager should Done

```
-- 11 --
610 • delete from employees where id = 10700;
611
612 • delete from employees where id = 10400;
613
614
615
616
```

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content: I

| comp_id | component_name | version | component_size | prog_language | comp_owner | component_status |
|---------|-------------------------|---------|----------------|---------------|------------|------------------|
| 1 | Keyboard Driver | K11 | 1200 | C | 10100 | Readv |
| 2 | Touch Screen Driver | T00 | 4000 | C++ | 10100 | Readv |
| 3 | Dbase Interface | D00 | 2500 | C++ | 10200 | Readv |
| 4 | Dbase Interface | D01 | 2500 | C++ | 10300 | Readv |
| 5 | Chart generator | C11 | 6500 | java | 10200 | not-readv |
| 6 | Pen Driver | P01 | 3575 | C | 10700 | not-readv |
| 7 | Math unit | A01 | 5000 | C | 10200 | usable |
| 8 | Math unit | A02 | 3500 | Java | 10200 | Readv |
| 9 | Dynamic Table Interface | D01 | 775 | JavaScript | 10400 | not-readv |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

components 22 x Apply Revert

Output

Action Output

| # | Time | Action | Message |
|-------|----------|------------------------------------------------------------------------------------------|-------------------|
| ✓ 283 | 15:28:04 | insert into Components(comp_id, component_name, version, component_size, prog_languag... | 1 row(s) affected |
| ✓ 284 | 15:28:43 | select * from components LIMIT 0, 10000 | 9 row(s) returned |

B) What is the Excel 2018beta product status?

select * from software_products where software_products.name like 'Excel' and software_products.version like '2018beta';

The screenshot shows a database management interface with a SQL editor and a results pane. The SQL editor contains a query to insert data into the 'software_products' table. The results pane shows a table with three columns: 'name', 'version', and 'software_status'. The first row of data is 'Excel', '2018beta', and 'not-readv'. The second row of data is 'NULL', 'NULL', and 'NULL'.

Query 1 fodm_project x

Find | Manager should

```
595 • insert into programming_language values('JavaScript','Current');
596 • insert into Components(comp_id, component_name, version, component_size, prog_language, comp_owner) values(9, '
597
598
599 -- 9B --
600 • * software_products software_products. 'Excel' software_products.version
601
602
603 -- 10 A --
604 -- Demonstrate the adding of an inspection on the component you just added. This inspection occurred on 11/20/2
605 • insert into inspection(inspection_id, component_name, version, inspection_date, by_who, score, description) val
606
607
608
609 -- 11 --
610 • delete from employees where id = 10700;
```

Result Grid

| name | version | software_status |
|-------|----------|-----------------|
| Excel | 2018beta | not-readv |
| NULL | NULL | NULL |

software_products23 x

Output

Action Output

| # | Time | Action | Message |
|-------|----------|-----------------------------------------------------------------------------------------------|-------------------|
| ✓ 284 | 15:28:43 | select * from components LIMIT 0, 10000 | 9 row(s) returned |
| ✓ 285 | 15:30:02 | select * from software_products where software_products.name like 'Excel' and software_pro... | 1 row(s) returned |

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".

insert into inspection(inspection_id, component_name, version, inspection_date, by_who, score, description) values(14, "Dynamic Table Interface", "D01", 11/20/2017, 10500, 80, "minor fixes needed");

Query 1 fodm_project x

Find Q | - Manager should Done

```

603 -- 10 A --
604 -- Demonstrate the adding of an inspection on the component you just added. This inspection occurred on 11/20/2
605 inspection(inspection_id, component_name, version, inspection_date, by_who, score, description)
606
607
608
609 -- 11 --

```

Result Grid

| inspection_id | component_name | version | inspection_date | by_who | score | description | status |
|---------------|-----------------------|---------|---------------------|--------|-------|---------------------------------------------------|-----------|
| 1 | Keyboard Driver | K11 | 2010-02-14 00:00:00 | 10100 | 100 | legacy code which is already approved | Ready |
| 2 | Touch Screen Driver | T00 | 2017-06-01 00:00:00 | 10200 | 95 | initial release ready for usage | Ready |
| 3 | Dbase Interface | D00 | 2010-02-22 00:00:00 | 10100 | 55 | too many hard coded parameters. the software... | not-ready |
| 4 | Dbase Interface | D00 | 2010-02-24 00:00:00 | 10100 | 78 | improved, but only handles DB2 format | usable |
| 5 | Dbase Interface | D00 | 2010-02-26 00:00:00 | 10100 | 95 | Okav. handles DB3 format. | Ready |
| 6 | Dbase Interface | D00 | 2010-02-28 00:00:00 | 10100 | 100 | satisfied | Ready |
| 7 | Dbase Interface | D01 | 2011-05-01 00:00:00 | 10200 | 100 | Okav ready for use | Ready |
| 8 | Pen Driver | P01 | 2017-07-15 00:00:00 | 10300 | 80 | Okav ready for beta testing | usable |
| 9 | Math unit | A01 | 2014-06-10 00:00:00 | 10100 | 90 | almost ready | usable |
| 10 | Math unit | A02 | 2014-06-15 00:00:00 | 10100 | 70 | Accuracy problems! | not-ready |
| 11 | Math unit | A02 | 2014-06-30 00:00:00 | 10100 | 100 | Okav problems fixed | Ready |
| 12 | Math unit | A02 | 2016-11-02 00:00:00 | 10700 | 100 | re-review for new employee to gain experience ... | Ready |
| 13 | Pen Driver | P01 | 0000-00-00 00:00:00 | 10400 | 60 | needs rework. introduced new errors | not-ready |
| 14 | Dynamic Table Inte... | D01 | 0000-00-00 00:00:00 | 10500 | 80 | minor fixes needed | usable |
| | | | | | | | |

inspection 24 x Apply Revert

Output

Action Output

| # | Time | Action | Message |
|-----|----------|-----------------------------------------------------------------------------------------------|--------------------|
| 286 | 15:31:00 | insert into inspection(inspection_id, component_name, version, inspection_date, by_who, sc... | 1 row(s) affected |
| 287 | 15:31:07 | select * from inspection LIMIT 0, 10000 | 14 row(s) returned |

B) What is the Excel 2018beta product status?

select * from software_products where software_products.name like 'Excel' and software_products.version like '2018beta';

The screenshot shows a database management interface with a query editor and a results pane. The query editor contains the following SQL code:

```
-- 9B --
select * from software_products where software_products.name like 'Excel' and software_products.version like '2
-- 10 A --
-- Demonstrate the adding of an inspection on the component you just added. This inspection occurred on 11/20/2
insert into inspection(inspection_id, component_name, version, inspection_date, by_who, score, description) val
-- 10 B --
select * from software_products where software_products.name like 'Excel' and software_products.version like '2
-- 11 --
delete from employees where id = 10700;
delete from employees where id = 10400;
```

The results pane shows a table with the following data:

| name | version | software_status |
|-------|----------|-----------------|
| Excel | 2018beta | not-readv |

The output pane shows the following actions and messages:

| # | Time | Action | Message |
|-----|----------|-----------------------------------------------------------------------------------------------|--------------------|
| 287 | 15:31:07 | select * from inspection LIMIT 0, 10000 | 14 row(s) returned |
| 288 | 15:32:10 | select * from software_products where software_products.name like 'Excel' and software_pro... | 1 row(s) returned |

GRADUATE:

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

Assumption:

1. When an employee leaves the company, it's work on the project should not be deleted. I.e. the components the employee has made or the inspection the employee has performed on any project.

So I have created a new table `employeeLeftInformation`. In this table all the employees that leave the company, their basic information like Employee ID and Name is stored in the table.

If the employee is a manager for other employees then, all the CURRENT employees with that manager would be set to null and the manager field would be updated when the new manager is assigned.

So even when component table is updated and if `comp_owner` and `by_who` is changed it will check both the tables to check if the data entered is correct or wrongly entered. A trigger will handle this.

`delete from employees where id = 10700;`

`select * from Employees;`

The screenshot shows a database management tool interface. At the top, there's a 'Query 1' tab with a 'fodm_project' window. Below the toolbar, a 'Find' search bar is visible. The main query editor contains the following SQL code:

```
639 select * from inspection;
640 select * from components_in_software_products;
641 select * from components;
642 select * from employees;
643 select * from software_products;
644 select * from programming_language;
645 select * from employeeLeftInformation;
646
647
648
649
```

Below the query editor, the 'Result Grid' is displayed, showing a table with 5 columns: `id`, `name`, `hire_date`, `mgr_id`, and `seniority`. The table contains 8 rows of employee data:

| id | name | hire_date | mgr_id | seniority |
|-------|------------|---------------------|--------|-----------|
| 10100 | Employee-1 | 1984-08-11 00:00:00 | 10100 | senior |
| 10200 | Employee-2 | 1994-08-11 00:00:00 | 10100 | senior |
| 10300 | Employee-3 | 2004-08-11 00:00:00 | 10200 | senior |
| 10400 | Employee-4 | 2008-01-11 00:00:00 | 10200 | senior |
| 10500 | Employee-5 | 2015-01-11 00:00:00 | 10400 | junior |
| 10600 | Employee-6 | 2015-01-11 00:00:00 | 10400 | junior |
| 10800 | Employee-8 | 2017-01-11 00:00:00 | 10200 | newbie |
| NULL | NULL | NULL | NULL | NULL |

Below the result grid, there's an 'employees 51' tab. At the bottom, the 'Output' window shows the execution results of the queries:

| # | Time | Action | Message |
|-----|----------|----------------------------------------|-------------------|
| 834 | 17:06:12 | delete from employees where id = 10700 | 1 row(s) affected |
| 835 | 17:06:21 | select * from employees LIMIT 0, 10000 | 7 row(s) returned |

select * from employeeLeftInformation

The screenshot shows a database query tool interface. At the top, there's a tab labeled "Query 1" and "fodm_project". Below the tab is a toolbar with various icons and a "Limit to 10000 rows" dropdown. A "Find" bar is present with a search icon and a "Done" button. The main area displays a SQL query with line numbers 635 to 646. The query is a series of SELECT statements from various tables, ending with "select * from employeeLeftInformation;". Below the query editor is a "Result Grid" section. It has a "Filter Rows:" input field and buttons for "Edit", "Export/Import", and "Wrap Cell Content:". The result grid shows a table with columns "id", "name", and "leftDate". The first row has values "10700", "Employee-7", and "2017-12-06 00:00:00". Below the result grid is a "Form Editor" button. At the bottom, there's an "Output" section with a dropdown menu set to "Action Output". It displays a log of actions with columns "#", "Time", "Action", and "Message". Two actions are listed: one at 17:06:21 for "select * from employees LIMIT 0, 10000" returning 7 rows, and another at 17:07:46 for "select * from employeeLeftInformation LIMIT 0, 10000" returning 1 row.

Query 1 fodm_project

Limit to 10000 rows

Find

635
636
637
638
639 • select * from inspection;
640 • select * from components_in_software_products;
641 • select * from components;
642 • select * from employees;
643 • select * from software_products;
644 • select * from programming_language;
645 • select * from employeeLeftInformation;
646

Result Grid

| id | name | leftDate |
|-------|------------|---------------------|
| 10700 | Employee-7 | 2017-12-06 00:00:00 |
| NULL | NULL | NULL |

Form Editor

employeeLeftInformation 52

Apply Revert

Output

Action Output

| # | Time | Action | Message |
|-----|----------|------------------------------------------------------|-------------------|
| 835 | 17:06:21 | select * from employees LIMIT 0, 10000 | 7 row(s) returned |
| 836 | 17:07:46 | select * from employeeLeftInformation LIMIT 0, 10000 | 1 row(s) returned |

select * from Components;

Query 1 fodm_project x

Find - insert into

```
632
633 • delete from employees where id = 10700;
634 -- delete from employees where id = 10400;
635
636
637
638
639 • select * from inspection;
640 • select * from components_in_software_products;
641 • select * from components;
642 • select * from employees;
643 • select * from software_products;
```

Result Grid

| comp_id | component_name | version | component_size | prog_language | comp_owner | component_status |
|---------|-------------------------|---------|----------------|---------------|------------|------------------|
| 1 | Keyboard Driver | K11 | 1200 | C | 10100 | Readv |
| 2 | Touch Screen Driver | T00 | 4000 | C++ | 10100 | Readv |
| 3 | Dbase Interface | D00 | 2500 | C++ | 10200 | Readv |
| 4 | Dbase Interface | D01 | 2500 | C++ | 10300 | Readv |
| 5 | Chart oenerator | C11 | 6500 | java | 10200 | not-readv |
| 6 | Pen Driver | P01 | 3575 | C | 10700 | not-readv |
| 7 | Math unit | A01 | 5000 | C | 10200 | usable |
| 8 | Math unit | A02 | 3500 | Java | 10200 | Readv |
| 9 | Dvnamic Table Interface | D01 | 775 | JavaScript | 10400 | usable |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

components 53 x

Output

| # | Time | Action | Message |
|-------|----------|------------------------------------------------------|-------------------|
| ✓ 836 | 17:07:46 | select * from employeeLeftInformation LIMIT 0, 10000 | 1 row(s) returned |
| ✓ 837 | 17:08:25 | select * from components LIMIT 0, 10000 | 9 row(s) returned |

select * from Inspection;

Query 1 fodm_project x

Limit to 10000 rows

Find: -- 11 --
 630
 631 -- 11 --
 632
 633 • delete from employees where id = 10700;
 634 -- delete from employees where id = 10400;
 635

Result Grid

| inspection_id | component_name | version | inspection_date | by_who | score | description | status |
|---------------|-----------------------|---------|---------------------|--------|-------|---------------------------------------------------|-----------|
| 1 | Keyboard Driver | K11 | 2010-02-14 00:00:00 | 10100 | 100 | legacy code which is already approved | ready |
| 2 | Touch Screen Driver | T00 | 2017-06-01 00:00:00 | 10200 | 95 | initial release | ready |
| 3 | Dbase Interface | D00 | 2010-02-22 00:00:00 | 10100 | 55 | too many hard coded parameters. the software... | not-ready |
| 4 | Dbase Interface | D00 | 2010-02-24 00:00:00 | 10100 | 78 | improved. but only handles DB2 format | usable |
| 5 | Dbase Interface | D00 | 2010-02-26 00:00:00 | 10100 | 95 | Okav. handles DB3 format. | Ready |
| 6 | Dbase Interface | D00 | 2010-02-28 00:00:00 | 10100 | 100 | satisfied | Ready |
| 7 | Dbase Interface | D01 | 2011-05-01 00:00:00 | 10200 | 100 | Okav ready for use | Ready |
| 8 | Pen Driver | P01 | 2017-07-15 00:00:00 | 10300 | 80 | Okav ready for beta testing | usable |
| 9 | Math unit | A01 | 2014-06-10 00:00:00 | 10100 | 90 | almost ready | usable |
| 10 | Math unit | A02 | 2014-06-15 00:00:00 | 10100 | 70 | Accuracy problems! | not-ready |
| 11 | Math unit | A02 | 2014-06-30 00:00:00 | 10100 | 100 | Okav problems fixed | Ready |
| 12 | Math unit | A02 | 2016-11-02 00:00:00 | 10700 | 100 | re-review for new employee to gain experience ... | Ready |
| 13 | Pen Driver | P01 | 0000-00-00 00:00:00 | 10400 | 60 | needs rework. introduced new errors | not-ready |
| 14 | Dynamic Table Inte... | D01 | 0000-00-00 00:00:00 | 10500 | 80 | minor fixes needed | usable |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL |

inspection 54 x Apply Revert

Output

Action Output

| # | Time | Action | Message |
|-----|----------|-----------------------------------------|--------------------|
| 837 | 17:08:25 | select * from components LIMIT 0, 10000 | 9 row(s) returned |
| 838 | 17:09:01 | select * from inspection LIMIT 0, 10000 | 14 row(s) returned |

To make my design show proof I have also tried to delete another employee with ID = 10400 who is a manager to few employees.

delete from employees where id = 10400;

select * from employees;

The screenshot shows a database management tool interface. At the top, there's a query editor with a toolbar and a search bar. Below the editor, a 'Result Grid' displays a table of employee data. The table has columns: id, name, hire_date, mgr_id, and seniority. The data shows several employees, with mgr_id values of 10100, 10200, and NULL. A 'junior' label is visible next to the row with mgr_id 10200. On the right side, there's a sidebar with buttons for 'Result Grid', 'Form Editor', 'Field Types', and 'Query Stats'. At the bottom, an 'Output' pane shows a log of actions, including a delete operation and a select operation.

| id | name | hire_date | mgr_id | seniority |
|-------|------------|---------------------|--------|-----------|
| 10100 | Emolovee-1 | 1984-08-11 00:00:00 | 10100 | senior |
| 10200 | Emolovee-2 | 1994-08-11 00:00:00 | 10100 | senior |
| 10300 | Emolovee-3 | 2004-08-11 00:00:00 | 10200 | senior |
| 10500 | Emolovee-5 | 2015-01-11 00:00:00 | NULL | junior |
| 10600 | Emolovee-6 | 2015-01-11 00:00:00 | NULL | junior |
| 10800 | Emolovee-8 | 2017-01-11 00:00:00 | 10200 | newbie |
| NULL | NULL | NULL | NULL | NULL |

employees 55 x

Output

| # | Time | Action | Message |
|-----|----------|----------------------------------------|-------------------|
| 839 | 17:10:15 | delete from employees where id = 10400 | 1 row(s) affected |
| 840 | 17:10:22 | select * from employees LIMIT 0, 10000 | 6 row(s) returned |

The employee with ID = 10400 was a manager to 2 employees. When he left the manager ID for other employees will be set to null until a new manager is assigned.

select * from employeeLeftInformation;

The screenshot shows a database query tool interface. At the top, there's a tab labeled "Query 1" and "fodm_project". Below the tab is a toolbar with various icons and a "Limit to 10000 rows" dropdown. A "Find" bar is also present. The main area displays a SQL query with line numbers 637 to 649. The query is:

```
637
638
639 • select * from inspection;
640 • select * from components_in_software_products;
641 • select * from components;
642 • select * from employees;
643 • select * from software_products;
644 • select * from programming_language;
645 • select * from employeeLeftInformation;
646
647
648
649
```

Below the query editor is a "Result Grid" section. It has a "Filter Rows:" input field and buttons for "Edit", "Export/Import", and "Wrap Cell Content". The grid displays the following data:

| | id | name | leftDate |
|--|-------|------------|---------------------|
| | 10400 | Employee-4 | 2017-12-06 00:00:00 |
| | 10700 | Employee-7 | 2017-12-06 00:00:00 |
| | NULL | NULL | NULL |

On the right side of the "Result Grid" section, there are buttons for "Result Grid" and "Form Editor". Below this is a tab labeled "employeeLeftInformation 56" with "Apply" and "Revert" buttons.

At the bottom, there's an "Output" section with a dropdown menu set to "Action Output". It displays a log of actions:

| # | Time | Action | Message |
|-------|----------|------------------------------------------------------|-------------------|
| ✓ 840 | 17:10:22 | select * from employees LIMIT 0, 10000 | 6 row(s) returned |
| ✓ 841 | 17:11:38 | select * from employeeLeftInformation LIMIT 0, 10000 | 2 row(s) returned |

select * from Components;

The screenshot shows a database management interface with a query editor and a results grid. The query editor contains the following SQL code:

```
630
631 -- 11 --
632
633 • delete from employees where id = 10700;
634 -- delete from employees where id = 10400;
635
636
637
638
639 • select * from inspection;
640 • select * from components in software products;
```

The results grid displays a table with the following columns: comp_id, component_name, version, component_size, prog_language, comp_owner, and component_status. The table contains 9 rows of data, with the last row highlighted in blue.

| comp_id | component_name | version | component_size | prog_language | comp_owner | component_status |
|---------|-------------------------|---------|----------------|---------------|------------|------------------|
| 1 | Keyboard Driver | K11 | 1200 | C | 10100 | Readv |
| 2 | Touch Screen Driver | T00 | 4000 | C++ | 10100 | Readv |
| 3 | Dbase Interface | D00 | 2500 | C++ | 10200 | Readv |
| 4 | Dbase Interface | D01 | 2500 | C++ | 10300 | Readv |
| 5 | Chart generator | C11 | 6500 | java | 10200 | not-readv |
| 6 | Pen Driver | P01 | 3575 | C | 10700 | not-readv |
| 7 | Math unit | A01 | 5000 | C | 10200 | usable |
| 8 | Math unit | A02 | 3500 | Java | 10200 | Readv |
| 9 | Dvnamic Table Interface | D01 | 775 | JavaScript | 10400 | usable |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

The output section shows the following actions:

| # | Time | Action | Message |
|-----|----------|------------------------------------------------------|-------------------|
| 841 | 17:11:38 | select * from employeeLeftInformation LIMIT 0, 10000 | 2 row(s) returned |
| 842 | 17:12:18 | select * from components LIMIT 0, 10000 | 9 row(s) returned |

The work that the employee Id 10400 did is not deleted as it is documented and should not be deleted.

select * from inspection;

Query 1 fodm_project x

Limit to 10000 rows

Find Q | -insert into Done

```
631 -- 11 --
632
633 delete from employees where id = 10700;
634 -- delete from employees where id = 10400;
```

Result Grid

| inspection_id | component_name | version | inspection_date | by_who | score | description | status |
|---------------|-----------------------|---------|---------------------|--------|-------|---------------------------------------------------|-----------|
| 1 | Keyboard Driver | K11 | 2010-02-14 00:00:00 | 10100 | 100 | legacy code which is already approved | Ready |
| 2 | Touch Screen Driver | T00 | 2017-06-01 00:00:00 | 10200 | 95 | initial release ready for usage | Ready |
| 3 | Dbase Interface | D00 | 2010-02-22 00:00:00 | 10100 | 55 | too many hard coded parameters, the software... | not-ready |
| 4 | Dbase Interface | D00 | 2010-02-24 00:00:00 | 10100 | 78 | improved, but only handles DB2 format | usable |
| 5 | Dbase Interface | D00 | 2010-02-26 00:00:00 | 10100 | 95 | Okav, handles DB3 format. | Ready |
| 6 | Dbase Interface | D00 | 2010-02-28 00:00:00 | 10100 | 100 | satisfied | Ready |
| 7 | Dbase Interface | D01 | 2011-05-01 00:00:00 | 10200 | 100 | Okav ready for use | Ready |
| 8 | Pen Driver | P01 | 2017-07-15 00:00:00 | 10300 | 80 | Okav ready for beta testing | usable |
| 9 | Math unit | A01 | 2014-06-10 00:00:00 | 10100 | 90 | almost ready | usable |
| 10 | Math unit | A02 | 2014-06-15 00:00:00 | 10100 | 70 | Accuracy problems! | not-ready |
| 11 | Math unit | A02 | 2014-06-30 00:00:00 | 10100 | 100 | Okav problems fixed | Ready |
| 12 | Math unit | A02 | 2016-11-02 00:00:00 | 10700 | 100 | re-review for new employee to gain experience ... | Ready |
| 13 | Pen Driver | P01 | 0000-00-00 00:00:00 | 10400 | 60 | needs rework, introduced new errors | not-ready |
| 14 | Dynamic Table Inte... | D01 | 0000-00-00 00:00:00 | 10500 | 80 | minor fixes needed | usable |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL |

inspection 58 x Apply Revert

Output

Action Output

| # | Time | Action | Message |
|-----|----------|-----------------------------------------|--------------------|
| 842 | 17:12:18 | select * from components LIMIT 0, 10000 | 9 row(s) returned |
| 843 | 17:13:24 | select * from inspection LIMIT 0, 10000 | 14 row(s) returned |

(END)