

Assignment 2 : SQL Views

Name: Muhammad Shahzaib Waseem

Class: BSCS-6B

Registration Number: 187330

Task1

CSStudents View

Code

USE DataBaseLab;

```
CREATE VIEW CSStudents AS  
SELECT * FROM Student  
WHERE major = "Computer Science";
```

Result

The screenshot displays the MySQL Workbench interface. The 'Query' tab is active, showing the SQL code for creating the 'CSStudents' view. The 'Result Grid' shows the output of the view, displaying student information for the 'Computer Science' major. The 'Output' tab at the bottom shows the execution log, including the creation of the view and the execution of a query that returns 4 rows.

snm	sname	major	level	age
112348546	Joseph Thomason	Computer Science	SO	19
115987938	Christopher Garcia	Computer Science	JR	20
322654189	Lisa Walker	Computer Science	SO	17
348121549	Paul Hall	Computer Science	JR	18

#	Time	Action	Message	Duration / Fetch
35	20:23:51	SELECT sname, age FROM CSStudents WHERE age IN (SELECT MAX(age) FROM CSStudents) LIMIT 0, 1...	1 row(s) returned	0.000 sec / 0.000 sec
36	20:24:16	USE DataBaseLab	0 row(s) affected	0.000 sec
37	20:24:16	DROP VIEW CSStudents	0 row(s) affected	0.000 sec
38	20:24:16	CREATE VIEW CSStudents AS SELECT * FROM Student WHERE major = "Computer Science"	0 row(s) affected	0.016 sec
39	20:24:16	SELECT MAX(age) AS OldestStudent FROM CSStudents LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
40	20:24:44	SELECT * FROM databaselab.csstudents LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec

Task A

Code

```
USE DataBaseLab;
```

```
DROP VIEW CSStudents;
```

```
CREATE VIEW CSStudents AS  
    SELECT * FROM Student  
    WHERE major = "Computer Science";
```

```
/* TASK A */  
SELECT MAX(age) AS OldestStudent FROM CSStudents;
```

Result

The screenshot displays the MySQL Workbench interface. The main editor window shows a series of SQL tasks: `USE DataBaseLab;`, `DROP VIEW CSStudents;`, `CREATE VIEW CSStudents AS SELECT * FROM Student WHERE major = "Computer Science";`, and `/* TASK A */ SELECT MAX(age) AS OldestStudent FROM CSStudents;`. The left sidebar shows the 'SCHEMAS' panel with 'databaseLab' selected, containing tables like 'enrolledstudents' and 'elderstudents'. The bottom panel shows the 'Output' window with a table of execution results.

#	Time	Action	Message	Duration / Fetch
34	20:23:51	CREATE VIEW CSStudents AS SELECT * FROM Student WHERE major = "Computer Science"	0 row(s) affected	0.016 sec
35	20:23:51	SELECT name, age FROM CSStudents WHERE age IN (SELECT MAX(age) FROM CSStudents) LIMIT 0, 1...	1 row(s) returned	0.000 sec / 0.000 sec
36	20:24:16	USE DataBaseLab	0 row(s) affected	0.000 sec
37	20:24:16	DROP VIEW CSStudents	0 row(s) affected	0.000 sec
38	20:24:16	CREATE VIEW CSStudents AS SELECT * FROM Student WHERE major = "Computer Science"	0 row(s) affected	0.016 sec
39	20:24:16	SELECT MAX(age) AS OldestStudent FROM CSStudents LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Task B

Code

```
USE DataBaseLab;
```

```
DROP VIEW CSStudents;
```

```
CREATE VIEW CSStudents AS  
    SELECT * FROM Student  
    WHERE major = "Computer Science";
```

```
/* TASK B */
```

```
SELECT sname, age FROM CSStudents  
WHERE age IN  
(  
    SELECT MAX(age) FROM CSStudents  
);
```

Result

The screenshot displays the MySQL Workbench interface. The central editor shows a series of SQL tasks: using the database, dropping the view, creating the view, and a task separator. Below the editor, the 'Result Grid' shows the output of the view query, displaying two rows: 'sname' and 'age' for 'Christopher Garcia' with an age of 20. The bottom 'Output' pane shows the execution log with timestamps and messages for each task.

#	Time	Action	Message	Duration / Fetch
30	20:23:28	CREATE VIEW CSStudents AS SELECT * FROM Student WHERE major = "Computer Science"	0 row(s) affected	0.016 sec
31	20:23:28	SELECT sname, age FROM CSStudents, Enrolled, Faculty, Class WHERE level = "JR" AND CSStude...	2 row(s) returned	0.000 sec / 0.000 sec
32	20:23:51	USE DataBaseLab	0 row(s) affected	0.000 sec
33	20:23:51	DROP VIEW CSStudents	0 row(s) affected	0.000 sec
34	20:23:51	CREATE VIEW CSStudents AS SELECT * FROM Student WHERE major = "Computer Science"	0 row(s) affected	0.016 sec
35	20:23:51	SELECT sname, age FROM CSStudents WHERE age IN (SELECT MAX(age) FROM CSStudents) LIMIT 0, 1...	1 row(s) returned	0.000 sec / 0.000 sec

Task C

Code

USE DataBaseLab;

```
/*Drop this View Everytime so there is no "Already Exists ERROR"*/  
DROP VIEW CSStudents;
```

```
CREATE VIEW CSStudents AS  
    SELECT * FROM Student  
    WHERE major = "Computer Science";
```

```
/* TASK C */
```

```
SELECT sname, major, age  
FROM CSStudents, Enrolled, Faculty, Class  
WHERE level = "JR"  
    AND CSStudents.snum = Enrolled.snum  
    AND Enrolled.cname = Class.cname  
    AND Class.fid = Faculty.fid  
    AND Faculty.fname = "Ivana Teach";
```

Result

The screenshot displays the MySQL Workbench interface. The central editor window shows a series of SQL commands being executed in a task window. The commands include using the database, dropping an existing view, creating a new view named CSStudents, and then querying it. The result grid shows the output of the final query, displaying three rows of student data.

sname	major	age
Christopher Garcia	Computer Science	20
Paul Hall	Computer Science	18

The bottom panel shows the 'Action Output' tab, which provides a detailed log of the executed statements and their results, including the number of rows affected and the duration of each operation.

Task D

Code

USE DataBaseLab;

```
/*Drop this View Everytime so there is no "Already Exists ERROR"*/  
DROP VIEW CSStudents;
```

```
CREATE VIEW CSStudents AS  
    SELECT * FROM Student  
    WHERE major = "Computer Science";
```

```
/* TASK C */
```

```
SELECT DISTINCT fname, deptid, Enrolled.cname, room  
FROM CSStudents, Enrolled, Faculty, Class  
WHERE Enrolled.snum = CSStudents.snum  
    AND Enrolled.cname = Class.cname  
    AND Class.fid = Faculty.fid;
```

Result

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

- Navigator:** Shows the database structure including 'databaseLab' with tables 'elderstudents', 'enrolledstudents', and 'Stored Procedures'.
- Task Editor:** Displays the SQL code for Task D, including the DROP VIEW, CREATE VIEW, and the final SELECT query.
- Result Grid:** Shows the output of the SELECT query, displaying 2 rows of data.
- Action Output:** Shows the execution log of the SQL tasks, including the time taken for each operation.

fname	deptid	cname	room
Ivana Teach	20	Database Systems	1320 DCL
Linda Davis	20	Operating System Design	20 ANW

#	Time	Action	Message	Duration / Fetch
22	20:22:16	SELECT * FROM databaseLab.elderstudents LIMIT 0, 1000	19 row(s) returned	0.000 sec / 0.000 sec
23	20:22:57	SELECT * FROM databaseLab.csstudents LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
24	20:23:09	USE DataBaseLab	0 row(s) affected	0.000 sec
25	20:23:09	DROP VIEW CSStudents	0 row(s) affected	0.000 sec
26	20:23:09	CREATE VIEW CSStudents AS SELECT * FROM Student WHERE major = "Computer Science"	0 row(s) affected	0.016 sec
27	20:23:09	SELECT DISTINCT fname, deptid, Enrolled.cname, room FROM CSStudents, Enrolled, Faculty, Class WHERE ...	2 row(s) returned	0.000 sec / 0.000 sec

Task 2

ElderStudents View

Code

USE DataBaseLab;

```
CREATE VIEW ElderStudents AS SELECT * FROM Student S1
WHERE age IN
(
    SELECT MAX(age) FROM Student S2
    WHERE S1.major = S2.major
    GROUP BY major
);
```

Result

The screenshot displays the MySQL Workbench interface. The 'Task2A' window shows the SQL editor with the following query:

```
SELECT * FROM databaselab.elderstudents;
```

The 'Result Grid' shows the output of the query, displaying a table with columns: snum, sname, major, level, and age. The data is as follows:

snum	sname	major	level	age
51135593	Maria White	English	SR	21
60839453	Charles Harris	Architecture	SR	22
99354543	Susan Martin	Law	JR	20
115987938	Christopher Garcia	Computer Science	JR	20
132977562	Ancela Martinez	History	SR	20
280158572	Maroaret Clark	Animal Science	FR	18
701791879	Juan Rodriguez	Psychology	SO	20

The 'Output' window shows the execution of the following SQL statements:

```
DROP VIEW ElderStudents
CREATE VIEW ElderStudents AS SELECT * FROM Student S1 WHERE age IN ( SELECT MAX(age) FROM ...
SELECT DISTINCT Class.cname, room FROM ElderStudents, Enrolled, Class WHERE ElderStudents.snum = ...
SELECT * FROM databaselab.elderstudents LIMIT 0, 1000
SELECT * FROM databaselab.elderstudents LIMIT 0, 1000
SELECT * FROM databaselab.elderstudents LIMIT 0, 1000
```

The 'Object Info' window shows the structure of the 'elderstudents' view:

```
Columns:
snum      int(11)
sname     varchar(30)
major     char(25)
level     char(2)
age       int(11)
```

Task A

Code

USE DataBaseLab;

DROP VIEW ElderStudents;

CREATE VIEW ElderStudents AS SELECT * FROM Student S1
WHERE age IN

```
(  
    SELECT MAX(age) FROM Student S2  
    WHERE S1.major = S2.major  
    GROUP BY major  
);
```

/* TASK A */

```
SELECT DISTINCT Class.cname, room  
FROM ElderStudents, Enrolled, Class  
WHERE ElderStudents.snum = Enrolled.snum  
AND Enrolled.cname = Class.cname;
```

Result

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' panel with a tree view showing the 'databaseLab' database and its tables. The main editor window shows the following SQL queries:

```
USE DataBaseLab;  
DROP VIEW ElderStudents;  
CREATE VIEW ElderStudents AS SELECT * FROM Student S1  
WHERE age IN (  
    SELECT MAX(age) FROM Student S2  
    WHERE S1.major = S2.major  
    GROUP BY major  
);  
/* TASK A */  
SELECT DISTINCT Class.cname, room  
FROM ElderStudents, Enrolled, Class  
WHERE ElderStudents.snum = Enrolled.snum  
AND Enrolled.cname = Class.cname;
```

The 'Result Grid' shows the output of the first query, displaying a list of classes and their rooms:

cname	room
Database Systems	1320 DCL
Operatino System Design	20 AVW
Communication Networks	20 AVW
Optical Electronics	R15
Perception	O3
Social Cognition	R15
Emergence Distributed Systems	20 AVW

The 'Output' panel at the bottom shows the execution log, including the following actions and messages:

#	Time	Action	Message	Duration / Fetch
14	20:21:09	CREATE VIEW ElderStudents AS SELECT * FROM Student S1 WHERE age IN (SELECT MAX(age) FROM ...	0 row(s) affected	0.015 sec
15	20:21:09	SELECT major, age ElderStudentAge FROM ElderStudents WHERE age IN (SELECT MAX(age) FROM Ede...	1 row(s) returned	0.016 sec / 0.000 sec
16	20:21:33	USE DataBaseLab	0 row(s) affected	0.000 sec
17	20:21:33	DROP VIEW ElderStudents	0 row(s) affected	0.000 sec
18	20:21:33	CREATE VIEW ElderStudents AS SELECT * FROM Student S1 WHERE age IN (SELECT MAX(age) FROM ...	0 row(s) affected	0.047 sec
19	20:21:33	SELECT DISTINCT Class.cname, room FROM ElderStudents, Enrolled, Class WHERE ElderStudents.snum = ...	10 row(s) returned	0.000 sec / 0.000 sec

Task B

Code

USE DataBaseLab;

DROP VIEW ElderStudents;

CREATE VIEW ElderStudents AS SELECT * FROM Student S1
WHERE age IN

```
(  
    SELECT MAX(age) FROM Student S2  
    WHERE S1.major = S2.major  
    GROUP BY major  
);
```

/* TASK B */

SELECT major, age EldestStudentAge FROM ElderStudents
WHERE age IN

```
(  
    SELECT MAX(age) FROM ElderStudents  
);
```

Result

The screenshot displays the MySQL Workbench interface. The central editor shows the SQL code for Task B, which includes dropping a view, creating a view named 'ElderStudents' based on a subquery, and then querying this view. The left sidebar shows the 'SCHEMAS' panel with 'databaseLab' selected, containing tables like 'enrolledstudents' and 'sakila'. The bottom panel shows the 'Action Output' log, detailing the execution of the SQL statements and their results.

#	Time	Action	Message	Duration / Fetch
43	20:27:06	CREATE VIEW ElderStudents AS SELECT * FROM Student S1 WHERE age IN (SELECT MAX(age) FROM ...	0 row(s) affected	0.032 sec
44	20:27:06	SELECT major, age EldestStudentAge FROM ElderStudents WHERE age IN (SELECT MAX(age) FROM Ede...	1 row(s) returned	0.000 sec / 0.000 sec
45	20:27:43	USE DataBaseLab	0 row(s) affected	0.000 sec
46	20:27:43	DROP VIEW ElderStudents	0 row(s) affected	0.000 sec
47	20:27:43	CREATE VIEW ElderStudents AS SELECT * FROM Student S1 WHERE age IN (SELECT MAX(age) FROM ...	0 row(s) affected	0.016 sec
48	20:27:43	SELECT major, age EldestStudentAge FROM ElderStudents WHERE age IN (SELECT MAX(age) FROM Ede...	1 row(s) returned	0.016 sec / 0.000 sec

Task 3 (EnrolledStudents View)

Code

USE DataBaseLab;

```
CREATE VIEW EnrolledStudents AS  
SELECT DISTINCT Student.snum, sname, major, level, age  
FROM Student, Enrolled  
WHERE Student.snum = Enrolled.snum;
```

Result

The screenshot displays the MySQL Workbench interface. The 'Query' tab is active, showing the SQL script for creating the 'EnrolledStudents' view. The script includes the following steps:

- USE DataBaseLab;
- DROP VIEW EnrolledStudents;
- CREATE VIEW EnrolledStudents AS
- SELECT DISTINCT Student.snum, sname, major, level, age
- FROM Student, Enrolled
- WHERE Student.snum = Enrolled.snum;
- SELECT * FROM databaselab.enrolledstudents;

The 'Result Grid' shows the output of the final SELECT statement, displaying 11 rows of student data. The 'Output' tab at the bottom shows the execution log, indicating that the view was successfully created and the final query returned 11 rows.

snum	sname	major	level	age
99354543	Susan Martin	Law	JR	20
112348546	Joseph Thomson	Computer Science	SO	19
115907938	Christopher Garcia	Computer Science	JR	20
301221823	Juan Rodriguez	Psychology	JR	20
322654189	Lisa Walker	Computer Science	SO	17
348121549	Paul Hall	Computer Science	JR	18
485768411	Luis Hernandez	Electrical Engineering	FR	17

Task 4

Code

USE DataBaseLab;

```
SELECT DISTINCT sname
FROM Student, Enrolled E1, Enrolled E2, Class C1, Class C2
WHERE Student.snum = E1.snum
      AND E1.cname = C1.cname
      AND E2.cname = C2.cname
      AND E1.cname != E2.cname
      AND C1.meets_at = C2.meets_at;
```

Result

The screenshot displays the MySQL Workbench interface. The central pane shows a SQL query in the editor, which is the same query provided in the 'Code' section. The query is executed, and the results are displayed in the 'Result Grid' pane below the editor. The results show a list of distinct student names (sname) from the 'Student' table, filtered by the specified conditions. The 'Action Output' pane at the bottom shows the execution progress, including the time taken for each step and the number of rows affected or returned.

#	Time	Action	Message	Duration / Fetch
1	20:18:52	USE DataBaseLab	0 row(s) affected	0.000 sec
2	20:18:52	DROP VIEW EnrolledStudents	0 row(s) affected	0.000 sec
3	20:18:52	CREATE VIEW EnrolledStudents AS SELECT DISTINCT Student.snum, sname, major, level, age FROM Student, Enrolled E1, Enrolled E2, Class C1, Class C2 WHERE E1.snum = E2.snum AND E1.cname = C1.cname AND E2.cname = C2.cname AND E1.cname != E2.cname AND C1.meets_at = C2.meets_at;	0 row(s) affected	0.015 sec
4	20:18:52	SELECT DISTINCT sname FROM EnrolledStudents, Enrolled E1, Enrolled E2, Class C1, Class C2 WHERE E1.snum = E2.snum AND E1.cname = C1.cname AND E2.cname = C2.cname AND E1.cname != E2.cname AND C1.meets_at = C2.meets_at;	0 row(s) returned	0.000 sec / 0.000 sec
5	20:19:28	USE DataBaseLab	0 row(s) affected	0.000 sec
6	20:19:28	SELECT DISTINCT sname FROM Student, Enrolled E1, Enrolled E2, Class C1, Class C2 WHERE Student.snum = E1.snum AND E1.cname = C1.cname AND E2.cname = C2.cname AND E1.cname != E2.cname AND C1.meets_at = C2.meets_at;	0 row(s) returned	0.000 sec / 0.000 sec

Task 5

Code

USE DataBaseLab;

DROP VIEW EnrolledStudents;

```
CREATE VIEW EnrolledStudents AS
SELECT DISTINCT Student.snum, sname, major, level, age
FROM Student, Enrolled
WHERE Student.snum = Enrolled.snum;
```

```
SELECT DISTINCT sname
FROM EnrolledStudents, Enrolled E1, Enrolled E2, Class C1, Class C2
WHERE E1.cname = C1.cname
    AND E1.snum = E2.snum
    AND E2.cname = C2.cname
    AND E1.cname != E2.cname
    AND C1.meets_at = C2.meets_at;
```

Result

The screenshot displays the MySQL Workbench interface for a local instance of MySQL 5.7. The central SQL editor contains four tasks:

1. USE DataBaseLab;
2. DROP VIEW EnrolledStudents;
3. CREATE VIEW EnrolledStudents AS
SELECT DISTINCT Student.snum, sname, major, level, age
FROM Student, Enrolled
WHERE Student.snum = Enrolled.snum;
4. SELECT DISTINCT sname
FROM EnrolledStudents, Enrolled E1, Enrolled E2, Class C1, Class C2
WHERE E1.cname = C1.cname
AND E1.snum = E2.snum
AND E2.cname = C2.cname
AND E1.cname != E2.cname
AND C1.meets_at = C2.meets_at;

The Action Output pane at the bottom shows the execution results:

#	Time	Action	Message	Duration / Fetch
1	20:18:52	USE DataBaseLab	0 row(s) affected	0.000 sec
2	20:18:52	DROP VIEW EnrolledStudents	0 row(s) affected	0.000 sec
3	20:18:52	CREATE VIEW EnrolledStudents AS SELECT DISTINCT Student.snum, sname, major, level, age FROM Student, Enrolled WHERE Student.snum = Enrolled.snum;	0 row(s) affected	0.015 sec
4	20:18:52	SELECT DISTINCT sname FROM EnrolledStudents, Enrolled E1, Enrolled E2, Class C1, Class C2 WHERE E1.cname = C1.cname AND E1.snum = E2.snum AND E2.cname = C2.cname AND E1.cname != E2.cname AND C1.meets_at = C2.meets_at;	0 row(s) returned	0.000 sec / 0.000 sec