

MySQL Workbench

University X

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

sys

SQL File 2

CREATE DATABASE University;

Output

Action Output

Time Action Message Duration / Fetch

15:00:15 CREATE DATABASE University 1 row(s) affected 0.000 sec

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

No object selected

Object Info Session

MySQL Workbench

University X

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

sys

university

Tables

Views

Stored Procedures

Functions

SQL File 2

CREATE DATABASE University;

USE University;

Output

Action Output

Time Action Message Duration / Fetch

15:00:15 CREATE DATABASE University 1 row(s) affected 0.000 sec

15:02:54 USE University 0 row(s) affected 0.000 sec

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

No object selected

Object Info Session

MySQL Workbench

University X

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

sys

university

Tables

Views

Stored Procedures

Functions

SQL File 2

CREATE DATABASE University;

USE University;

CREATE TABLE Students (

StudentID INT PRIMARY KEY,

Name VARCHAR(50),

Age INT,

Major VARCHAR(50))

Output

Action Output

Time Action Message Duration / Fetch

15:00:15 CREATE DATABASE University 1 row(s) affected 0.000 sec

15:02:54 USE University 0 row(s) affected 0.000 sec

15:04:11 CREATE TABLE Students (StudentID INT PRIMARY KEY, Name VARCHAR(50), Age INT, Major VA 0 row(s) affected 0.000 sec

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

No object selected

Object Info Session

MySQL Workbench - University

Schemas

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

SQL File 2*

```

2 USE University;
3 
4 CREATE TABLE Students (
5   StudentID INT PRIMARY KEY,
6   Name VARCHAR(50),
7   Age INT,
8   Major VARCHAR(50)
9 );
10 
11 
12 CREATE TABLE Courses (
13   CourseID INT PRIMARY KEY,
14   CourseName VARCHAR(50),
15   Credits INT
16 );
17 
```

Output

Action	Time	Action	Message	Duration / Fetch
1	15:02:15	CREATE DATABASE University	1 row(s) affected	0.000 sec
2	15:02:54	USE University	0 row(s) affected	0.000 sec
3	15:04:11	CREATE TABLE Students (StudentID INT PRIMARY KEY, Name VARCHAR(50), Age INT, Major VA...)	0 row(s) affected	0.032 sec
4	15:55:32	CREATE TABLE Courses (CourseID INT PRIMARY KEY, CourseName VARCHAR(50), Credits INT)	0 row(s) affected	0.047 sec

Object Info **Session**

MySQL Workbench - University

Schemas

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

SQL File 2*

```

1 CREATE TABLE Courses (
2   CourseID INT PRIMARY KEY,
3   CourseName VARCHAR(50),
4   Credits INT
5 );
6 
7 
8 CREATE TABLE Enrollments (
9   EnrollmentID INT PRIMARY KEY,
10  StudentID INT,
11  CourseID INT,
12  Grade CHAR(2),
13  FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
14  FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
15 );
16 
```

Output

Action	Time	Action	Message	Duration / Fetch
1	15:00:15	CREATE DATABASE University	1 row(s) affected	0.000 sec
2	15:02:54	USE University	0 row(s) affected	0.000 sec
3	15:04:11	CREATE TABLE Students (StudentID INT PRIMARY KEY, Name VARCHAR(50), Age INT, Major VA...)	0 row(s) affected	0.032 sec
4	15:55:32	CREATE TABLE Courses (CourseID INT PRIMARY KEY, CourseName VARCHAR(50), Credits INT)	0 row(s) affected	0.047 sec
5	15:57:47	CREATE TABLE Enrollments (EnrollmentID INT PRIMARY KEY, StudentID INT, CourseID INT, Grade ...)	0 row(s) affected	0.031 sec

Object Info **Session**

MySQL Workbench - University

Schemas

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

SQL File 2*

```

26 
27 INSERT INTO Students (StudentID, Name, Age, Major) VALUES
28 (1, 'Aarohi', 19, 'Data Science'),
29 (2, 'Praneeth', 21, 'Data Science'),
30 (3, 'Carlton', 20, 'Computer Science'),
31 (4, 'Divya', 24, 'Mathematics'),
32 (5, 'Elena', 18, 'Physics'),
33 (6, 'Farhan', 22, 'Computer Science'),
34 (7, 'Grace', 23, 'Undeclared'),
35 (8, 'Hars', 20, 'Computer Science'),
36 (9, 'Irene', 21, 'Mathematics'),
37 (10, 'Jayden', 24, 'Computer Science'),
38 
```

Output

Action	Time	Action	Message	Duration / Fetch
1	15:00:15	CREATE DATABASE University	1 row(s) affected	0.000 sec
2	15:02:54	USE University	0 row(s) affected	0.000 sec
3	15:04:11	CREATE TABLE Students (StudentID INT PRIMARY KEY, Name VARCHAR(50), Age INT, Major VA...)	0 row(s) affected	0.032 sec
4	15:55:32	CREATE TABLE Courses (CourseID INT PRIMARY KEY, CourseName VARCHAR(50), Credits INT)	0 row(s) affected	0.047 sec
5	15:57:47	CREATE TABLE Enrollments (EnrollmentID INT PRIMARY KEY, StudentID INT, CourseID INT, Grade ...)	0 row(s) affected	0.031 sec
6	16:06:16	INSERT INTO Students (StudentID, Name, Age, Major) VALUES (1, 'Aarohi', 19, 'Data Science'), (2, 'Praneeth', 21, 'Data Science')	15 rows(s) affected Records: 15 Duplicates: 0 Warnings: 0	0.016 sec

Object Info **Session**

MySQL Workbench - University

Navigator:

- SCHEMAS
 - sys
 - university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

SQL Editor 1:

```

39 (13, 'Shantu', 19, 'Data Science'),
40 (13, 'Maya', 23, 'Computer Science'),
41 (14, 'Neill', 17, 'Undeclared'),
42 (15, 'Omar', 20, 'Mathematics')
43
44 • SELECT * FROM Students
45

```

Result Grid:

StudentID	Name	Age	Major
1	Aarohi	19	Data Science
2	Praneeth	23	Computer Science
3	Carlos	20	Computer Science
4	Divya	24	Mathematics
5	Elena	18	Physics
6	Fahim	22	Computer Science
7	Grace	23	Undeclared
8	Harish	20	Computer Science
9	Irene	21	Mathematics
10	Jordan	22	Computer Science
11	Kara	22	Physics
12	Shantu	19	Data Science
13	Maya	23	Computer Science
14	Neill	17	Undeclared
15	Omar	20	Mathematics

SQL Additions:

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Object Info: Session

MySQL Workbench - University

Navigator:

- SCHEMAS
 - sys
 - university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

SQL Editor 2:

```

40 (13, 'Maya', 23, 'Computer Science'),
41 (14, 'Neill', 17, 'Undeclared'),
42 (15, 'Omar', 20, 'Mathematics')
43
44 • SELECT * FROM Students
45
46 • INSERT INTO Courses (CourseID, CourseName, Credits) VALUES
47 (101, 'Database Systems', 4),
48 (102, 'Algorithms', 4),
49 (103, 'Machine Learning', 3),
50 (104, 'Web Development', 3)
51
52

```

Output:

Action	Time	Message	Duration / Fetch
1 15:05:32 CREATE TABLE Courses (CourseID INT PRIMARY KEY , CourseName VARCHAR(50) , Credits INT)		0 row(s) affected	0.047 sec
2 15:07:47 CREATE TABLE Enrollments (EnrollmentID INT PRIMARY KEY , StudentID INT , CourseID INT , Grade INT)		0 row(s) affected	0.031 sec
3 16:06:16 INSERT INTO Students (StudentID, Name, Age, Major) VALUES (1, 'Aarohi', 19, 'Data Science'), (2, 'Praneeth', 23, 'Computer Science')	16:06:16	15 row(s) affected Records: 15 Duplicates: 0 Warnings: 0	0.016 sec
4 16:15:57 SELECT * FROM Students LIMIT 0, 1000		15 row(s) returned	0.000 sec / 0.000 sec
5 16:19:08 INSERT INTO Courses (CourseID, CourseName, Credits) VALUES (101, 'Database Systems', 4), (102, 'Algorithms', 4)	16:19:08	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0	0.000 sec

Context Help: Snippets

Object Info: Session

MySQL Workbench - University

Navigator:

- SCHEMAS
 - sys
 - university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

SQL Editor 3:

```

45
46 • INSERT INTO Courses (CourseID, CourseName, Credits) VALUES
47 (101, 'Database Systems', 4),
48 (102, 'Algorithms', 4),
49 (103, 'Machine Learning', 3),
50 (104, 'Web Development', 3)
51
52 • SELECT * FROM Courses

```

Result Grid:

CourseID	CourseName	Credits
101	Database Systems	4
102	Algorithms	4
103	Machine Learning	3
104	Web Development	3

SQL Additions:

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Object Info: Session

Output:

Action	Time	Message	Duration / Fetch
1 15:00:15 CREATE DATABASE University		1 row(s) affected	0.000 sec
2 15:02:54 USE University		0 row(s) affected	0.000 sec
3 15:04:11 CREATE TABLE Students (StudentID INT PRIMARY KEY , Name VARCHAR(50) , Age INT , Major VA...)		0 row(s) affected	0.032 sec
4 15:55:32 CREATE TABLE Courses (CourseID INT PRIMARY KEY , CourseName VARCHAR(50) , Credits INT)		0 row(s) affected	0.047 sec
5 15:57:47 CREATE TABLE Enrollments (EnrollmentID INT PRIMARY KEY , StudentID INT , CourseID INT , Grade INT)		0 row(s) affected	0.031 sec
6 16:06:16 INSERT INTO Students (StudentID, Name, Age, Major) VALUES (1, 'Aarohi', 19, 'Data Science'), (2, 'Praneeth', 23, 'Computer Science')	16:06:16	15 row(s) affected Records: 15 Duplicates: 0 Warnings: 0	0.016 sec
7 16:15:57 SELECT * FROM Students LIMIT 0, 1000		15 row(s) returned	0.000 sec / 0.000 sec
8 16:19:08 INSERT INTO Courses (CourseID, CourseName, Credits) VALUES (101, 'Database Systems', 4), (102, 'Algorithms', 4)	16:19:08	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0	0.000 sec
9 16:21:19 SELECT * FROM Courses LIMIT 0, 1000		4 row(s) returned	0.000 sec / 0.000 sec

MySQL Workbench

Schemas

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

SQL Editor 2

```

48   (102, 'Algorithms', 4),
49   (103, 'Machine Learning', 3),
50   (104, 'Web Development', 3));
51
52 • SELECT * FROM Courses;
53
54 • INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade) VALUES
55   (1, 1, 101, 'A'),
56   (2, 2, 103, 'B'),
57   (3, 3, 102, 'A'),
58   (4, 4, 103, 'C'),
59   (5, 1, 102, 'B'),
60   (6, 6, 101, 'A-'),
61   (7, 7, 103, 'B+'),
62   (8, 8, 102, 'C+'),
63   (9, 9, 103, 'A'),
64   (10, 2, 101, 'B'));

```

Output

Action	Time	Action	Message	Duration / Fetch
1	15:06:15	CREATE DATABASE University	1 row(s) affected	0.000 sec
2	15:02:54	USE University	0 row(s) affected	0.000 sec
3	15:04:11	CREATE TABLE Students (StudentID INT PRIMARY KEY, Name VARCHAR(50), Age INT, Major VA...	0 row(s) affected	0.032 sec
4	15:05:22	CREATE TABLE Courses (CourseID INT PRIMARY KEY, CourseName VARCHAR(50), Cred...	0 row(s) affected	0.047 sec
5	15:57:47	CREATE TABLE Enrollments (EnrollmentID INT PRIMARY KEY, StudentID INT, CourseID INT, Grade ...)	0 row(s) affected	0.031 sec
6	16:06:16	INSERT INTO Students (StudentID, Name, Age, Major) VALUES (1, 'Aarohi', 19, 'Data Science');	15 row(s) affected Records: 15 Duplicates: 0 Warnings: 0	0.016 sec
7	16:15:57	SELECT * FROM Students LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec
8	16:19:08	INSERT INTO Courses (CourseID, CourseName, Credits) VALUES (101, 'Database Systems', 4);	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0	0.000 sec
9	16:21:19	SELECT * FROM Courses LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
10	16:48:56	INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade) VALUES (1, 1, 101, 'A');	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec
11	17:08:35	SELECT * FROM Enrollments LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

MySQL Workbench

Schemas

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

SQL Editor 2

```

65 • SELECT * FROM Enrollments;
66
67 -- -----
68 -- Executing All Provided Queries
69
70 --

```

Result Grid

EnrollmentID	StudentID	CourseID	Grade
1	2	101	A
2	2	103	B
3	3	102	A
4	4	103	C
5	5	102	B
6	6	101	A-
7	7	101	B+
8	8	102	C+
9	9	103	A
10	10	101	B
*			

Output

Action	Time	Action	Message	Duration / Fetch
6	15:06:16	INSERT INTO Students (StudentID, Name, Age, Major) VALUES (1, 'Aarohi', 19, 'Data Science');	15 row(s) affected Records: 15 Duplicates: 0 Warnings: 0	0.016 sec
7	15:15:57	SELECT * FROM Students LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec
8	16:19:08	INSERT INTO Courses (CourseID, CourseName, Credits) VALUES (101, 'Database Systems', 4);	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0	0.000 sec
9	16:21:19	SELECT * FROM Courses LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
10	16:48:56	INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade) VALUES (1, 1, 101, 'A');	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec
11	17:08:35	SELECT * FROM Enrollments LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

MySQL Workbench

Schemas

- sys
- university
 - Tables
 - departments
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

SQL Editor 2

```

55   (1, 1, 101, 'A'),
56   (2, 2, 103, 'B'),
57   (3, 3, 102, 'A'),
58   (4, 4, 103, 'C'),
59   (5, 1, 102, 'B'),
60   (6, 6, 101, 'A-'),
61   (7, 7, 103, 'B+'),
62   (8, 8, 102, 'C+'),
63   (9, 9, 103, 'A'),
64   (10, 2, 101, 'B'));
65
66 • SELECT * FROM Enrollments;
67
68 -- -----
69 -- Executing All Provided Queries
70
71
72 -- Question: How do you define a temporary table for departments?
73 • CREATE TABLE Departments (
74   DeptID INT PRIMARY KEY,
75   DeptName VARCHAR(50),
76 );

```

Output

Action	Time	Action	Message	Duration / Fetch
7	16:15:57	SELECT * FROM Students LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec
8	16:19:08	INSERT INTO Courses (CourseID, CourseName, Credits) VALUES (101, 'Database Systems', 4);	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0	0.000 sec
9	16:21:19	SELECT * FROM Courses LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
10	16:48:56	INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade) VALUES (1, 1, 101, 'A');	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec
11	17:08:35	SELECT * FROM Enrollments LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
12	17:11:59	CREATE TABLE Departments (DeptID INT PRIMARY KEY, DeptName VARCHAR(50))	0 row(s) affected	0.031 sec

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

MySQL Workbench

University

Schemas: university

```

74   DeptID INT PRIMARY KEY,
75   DeptName VARCHAR(50)
76 )
77
78 -- Question: How can you add a new 'Email' column to the Students table?
79 • ALTER TABLE Students ADD Email VARCHAR(100);
80 • SELECT * FROM Students;

```

Result Grid | Filter Rows | Edit | Export/Import | Wrap Cell Contents | Result Set | Form Editor | Data Types | Query Stats | Context Help | Snippets

StudentID	Name	Age	Major	Email
1	Pranesh	21	Data Science	
2	Carlos	20	Computer Science	
3	Dave	21	Mathematics	
4	Elena	18	Physics	
5	Fahim	22	Computer Science	
6	Gina	23	Mathematics	
7	Hannah	20	Computer Science	
8	Irene	21	Mathematics	
9	Jason	20	Computer Science	
10	Kara	22	Physics	
11	Shanu	19	Data Science	
12	Maya	23	Computer Science	
13	Nick	17	Mathematics	
14	Omar	20	Mathematics	
15				

Action Output | Time Action | Message | Duration / Fetch

- 10 17:08:56 INSERT INTO Enrolments (EnrollmentID, StudentID, CourseID, Grade) VALUES (1, 1, 101, 'A');
- 11 17:08:56 SELECT * FROM Enrolments LIMIT 0, 1000;
- 12 17:11:59 CREATE TABLE Departments (DeptID INT PRIMARY KEY, DeptName VARCHAR(50))
- 13 17:18:20 ALTER TABLE Students ADD Email VARCHAR(100)
- 14 17:18:28 SELECT * FROM Students LIMIT 0, 1000

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or toggle automatic help.

MySQL Workbench

University

Schemas: university

```

68 -- Executing All Provided Queries
69
70
71
72 -- Question: How do you define a temporary table for departments?
73 • CREATE TABLE Departments (
74   DeptID INT PRIMARY KEY,
75   DeptName VARCHAR(50)
76 );
77
78 -- Question: How can you add a new 'Email' column to the Students table?
79 • ALTER TABLE Students ADD Email VARCHAR(100);
80 • SELECT * FROM Students;
81
82 -- Question: How do you completely remove the Departments table?
83 • DROP TABLE Departments;

```

Action Output | Time Action | Message | Duration / Fetch

- 4 15:55:32 CREATE TABLE Courses (CourseID INT PRIMARY KEY, CourseName VARCHAR(50), Credits INT) 0 rows affected 0.047 sec
- 5 15:57:47 CREATE TABLE Enrolments (EnrollmentID INT PRIMARY KEY, StudentID INT, CourseID INT, Grade VARCHAR(2)) 0 rows affected 0.031 sec
- 6 16:06:16 INSERT INTO Students (StudentID, Name, Age, Major) VALUES (1, 'Karan', 19, 'Data Science'); (2, Pranesh, 21, 'Computer Science'); (3, Carlos, 20, 'Computer Science'); (4, Dave, 21, 'Mathematics'); (5, Elena, 18, 'Physics'); (6, Fahim, 22, 'Computer Science'); (7, Gina, 23, 'Mathematics'); (8, Hannah, 20, 'Computer Science'); (9, Irene, 21, 'Mathematics'); (10, Jason, 20, 'Computer Science'); (11, Kara, 22, 'Physics'); (12, Shanu, 19, 'Data Science'); (13, Maya, 23, 'Computer Science'); (14, Nick, 17, 'Mathematics'); (15, Omar, 20, 'Mathematics') 15 rows affected Records: 15 Duplicates: 0 Warnings: 0 0.016 sec
- 7 16:15:57 SELECT * FROM Students LIMIT 0, 1000 15 rows returned 0.000 sec / 0.000 sec 0.016 sec
- 8 16:19:08 INSERT INTO Courses (CourseID, CourseName, Credits) VALUES (101, 'Database Systems', 4); (102, 'Algorithm', 4) 4 rows affected Records: 4 Duplicates: 0 Warnings: 0 0.000 sec 0.000 sec
- 9 16:21:19 SELECT * FROM Courses LIMIT 0, 1000 4 rows returned 0.000 sec / 0.000 sec 0.016 sec
- 10 16:48:56 INSERT INTO Enrolments (EnrollmentID, StudentID, CourseID, Grade) VALUES (1, 1, 101, 'A'); (2, 2, 103, 'B'); (3, 3, 104, 'C'); (4, 4, 105, 'D'); (5, 5, 106, 'E'); (6, 6, 107, 'F'); (7, 7, 108, 'G'); (8, 8, 109, 'H'); (9, 9, 110, 'I'); (10, 10, 111, 'J'); (11, 11, 112, 'K'); (12, 12, 113, 'L'); (13, 13, 114, 'M'); (14, 14, 115, 'N'); (15, 15, 116, 'O') 10 rows affected Records: 10 Duplicates: 0 Warnings: 0 0.016 sec
- 11 17:08:35 SELECT * FROM Enrolments LIMIT 0, 1000 10 rows returned 0.000 sec / 0.000 sec 0.016 sec
- 12 17:11:59 CREATE TABLE Departments (DeptID INT PRIMARY KEY, DeptName VARCHAR(50)) 0 rows affected 0.031 sec
- 13 17:18:20 ALTER TABLE Students ADD Email VARCHAR(100) 0 rows affected Records: 0 Duplicates: 0 Warnings: 0 0.031 sec
- 14 17:18:28 SELECT * FROM Students LIMIT 0, 1000 15 rows returned 0.000 sec / 0.000 sec 0.016 sec
- 15 17:20:22 DROP TABLE Departments 0 rows affected 0.016 sec
- 16 17:22:17 ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17) 15 rows affected Records: 15 Duplicates: 0 Warnings: 0 0.016 sec

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or toggle automatic help.

MySQL Workbench

University

Schemas: university

```

71
72 -- Question: How do you define a temporary table for departments?
73 • CREATE TABLE Departments (
74   DeptID INT PRIMARY KEY,
75   DeptName VARCHAR(50)
76 );
77
78 -- Question: How can you add a new 'Email' column to the Students table?
79 • ALTER TABLE Students ADD Email VARCHAR(100);
80 • SELECT * FROM Students;
81
82 -- Question: How do you completely remove the Departments table?
83 • DROP TABLE Departments;
84
85 -- Question: How do you add a rule to ensure students are at least 17?
86 • ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17);

```

Action Output | Time Action | Message | Duration / Fetch

- 5 15:57:47 CREATE TABLE Enrolments (EnrollmentID INT PRIMARY KEY, StudentID INT, CourseID INT, Grade VARCHAR(2)) 0 rows affected 0.031 sec
- 6 16:06:16 INSERT INTO Students (StudentID, Name, Age, Major) VALUES (1, 'Karan', 19, 'Data Science'); (2, Pranesh, 21, 'Computer Science'); (3, Carlos, 20, 'Computer Science'); (4, Dave, 21, 'Mathematics'); (5, Elena, 18, 'Physics'); (6, Fahim, 22, 'Computer Science'); (7, Gina, 23, 'Mathematics'); (8, Hannah, 20, 'Computer Science'); (9, Irene, 21, 'Mathematics'); (10, Jason, 20, 'Computer Science'); (11, Kara, 22, 'Physics'); (12, Shanu, 19, 'Data Science'); (13, Maya, 23, 'Computer Science'); (14, Nick, 17, 'Mathematics'); (15, Omar, 20, 'Mathematics') 15 rows affected Records: 15 Duplicates: 0 Warnings: 0 0.016 sec
- 7 16:15:57 SELECT * FROM Students LIMIT 0, 1000 15 rows returned 0.000 sec / 0.000 sec 0.016 sec
- 8 16:19:08 INSERT INTO Courses (CourseID, CourseName, Credits) VALUES (101, 'Database Systems', 4); (102, 'Algorithm', 4) 4 rows affected Records: 4 Duplicates: 0 Warnings: 0 0.000 sec 0.000 sec
- 9 16:21:19 SELECT * FROM Courses LIMIT 0, 1000 4 rows returned 0.000 sec / 0.000 sec 0.016 sec
- 10 16:48:56 INSERT INTO Enrolments (EnrollmentID, StudentID, CourseID, Grade) VALUES (1, 1, 101, 'A'); (2, 2, 103, 'B'); (3, 3, 104, 'C'); (4, 4, 105, 'D'); (5, 5, 106, 'E'); (6, 6, 107, 'F'); (7, 7, 108, 'G'); (8, 8, 109, 'H'); (9, 9, 110, 'I'); (10, 10, 111, 'J'); (11, 11, 112, 'K'); (12, 12, 113, 'L'); (13, 13, 114, 'M'); (14, 14, 115, 'N'); (15, 15, 116, 'O') 10 rows affected Records: 10 Duplicates: 0 Warnings: 0 0.016 sec
- 11 17:08:35 SELECT * FROM Enrolments LIMIT 0, 1000 10 rows returned 0.000 sec / 0.000 sec 0.016 sec
- 12 17:11:59 CREATE TABLE Departments (DeptID INT PRIMARY KEY, DeptName VARCHAR(50)) 0 rows affected 0.031 sec
- 13 17:18:20 ALTER TABLE Students ADD Email VARCHAR(100) 0 rows affected Records: 0 Duplicates: 0 Warnings: 0 0.031 sec
- 14 17:18:28 SELECT * FROM Students LIMIT 0, 1000 15 rows returned 0.000 sec / 0.000 sec 0.016 sec
- 15 17:20:22 DROP TABLE Departments 0 rows affected 0.016 sec
- 16 17:22:17 ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17) 15 rows affected Records: 15 Duplicates: 0 Warnings: 0 0.063 sec

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or toggle automatic help.

MySQL Workbench

University

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - views
 - Stored Procedures
 - Functions

SQL File 2*

```

77
78 -- Question: How can you add a new 'Email' column to the Students table?
79 • ALTER TABLE Students ADD Email VARCHAR(100);
80 • SELECT * FROM Students;
81
82 -- Question: How do you completely remove the Departments table?
83 • DROP TABLE Departments;
84
85 -- Question: How do you add a rule to ensure students are at least 17?
86 • ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17);
87
88 -- Question: How do you remove the age-checking rule?
89 • ALTER TABLE Students DROP CONSTRAINT AgeCheck;
90

```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	15:55:32	CREATE TABLE Courses (CourseID INT PRIMARY KEY, CourseName VARCHAR(50), Credits INT)	0 rows(a) affected	0.047 sec
2	16:00:16	INSERT INTO Students (StudentID, Name, Age, Major) VALUES ('1', 'Nariv', 19, 'Data Science');	0 rows(a) inserted	0.031 sec
3	16:18:57	SELECT * FROM Students;	15 rows(a) returned	0.016 sec
4	16:19:08	INSERT INTO Courses (CourseID, CourseName, Credits) VALUES ('101', 'Database Systems', 4);	0 rows(a) affected Records: 4 Duplicates: 0 Warnings: 0	0.000 sec / 0.000 sec
5	16:21:19	SELECT * FROM Courses LIMIT 0, 1000	4 rows(a) returned	0.000 sec / 0.000 sec
6	16:48:56	INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade) VALUES ('1', 101, '101', 'A');	10 rows(a) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec
7	17:08:35	SELECT * FROM Enrollments LIMIT 0, 1000	10 rows(a) returned	0.000 sec / 0.000 sec
8	17:11:59	CREATE TABLE Departments (DeptID INT PRIMARY KEY, DeptName VARCHAR(50))	0 rows(a) affected	0.031 sec
9	17:18:20	ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17);	0 rows(a) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
10	17:20:22	DROP TABLE Departments;	15 rows(a) returned	0.000 sec / 0.000 sec
11	17:22:17	ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17);	0 rows(a) affected Records: 15 Duplicates: 0 Warnings: 0	0.063 sec
12	17:25:20	ALTER TABLE Students DROP CONSTRAINT AgeCheck;	0 rows(a) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec
13	17:26:33	UPDATE Students SET Major = 'Data Science' WHERE StudentID = 1;	1 rows(a) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec
14	17:28:08	SELECT * FROM Students LIMIT 0, 1000	15 rows(a) returned	0.016 sec

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

MySQL Workbench

University

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - views
 - Stored Procedures
 - Functions

SQL File 2*

```

80 • SELECT * FROM Students;
81
82 -- Question: How do you completely remove the Departments table?
83 • DROP TABLE Departments;
84
85 -- Question: How do you add a rule to ensure students are at least 17?
86 • ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17);
87
88 -- Question: How do you remove the age-checking rule?
89 • ALTER TABLE Students DROP CONSTRAINT AgeCheck;
90
91 -- Question: How do you update the major for student with ID 1 to 'Data Science'?
92 • UPDATE Students SET Major = 'Data Science' WHERE StudentID = 1;
93

```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	15:57:47	CREATE TABLE Courses (CourseID INT PRIMARY KEY, CourseName VARCHAR(50), Credits INT)	0 rows(a) affected	0.031 sec
2	16:00:16	INSERT INTO Students (StudentID, Name, Age, Major) VALUES ('1', 'Nariv', 19, 'Data Science');	0 rows(a) inserted	0.031 sec
3	16:15:57	SELECT * FROM Students;	15 rows(a) returned	0.000 sec / 0.000 sec
4	16:19:08	INSERT INTO Courses (CourseID, CourseName, Credits) VALUES ('101', 'Database Systems', 4);	0 rows(a) affected Records: 4 Duplicates: 0 Warnings: 0	0.000 sec
5	16:21:19	SELECT * FROM Courses LIMIT 0, 1000	4 rows(a) returned	0.000 sec / 0.000 sec
6	16:48:56	INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade) VALUES ('1', 101, '101', 'A');	10 rows(a) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec
7	17:08:35	SELECT * FROM Enrollments LIMIT 0, 1000	10 rows(a) returned	0.000 sec / 0.000 sec
8	17:11:59	CREATE TABLE Departments (DeptID INT PRIMARY KEY, DeptName VARCHAR(50))	0 rows(a) affected	0.031 sec
9	17:18:20	ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17);	0 rows(a) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
10	17:20:22	DROP TABLE Departments;	15 rows(a) returned	0.000 sec / 0.000 sec
11	17:22:17	ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17);	0 rows(a) affected Records: 15 Duplicates: 0 Warnings: 0	0.063 sec
12	17:25:20	ALTER TABLE Students DROP CONSTRAINT AgeCheck;	0 rows(a) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec
13	17:26:33	UPDATE Students SET Major = 'Data Science' WHERE StudentID = 1;	1 rows(a) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec
14	17:28:08	SELECT * FROM Students LIMIT 0, 1000	15 rows(a) returned	0.016 sec

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

MySQL Workbench

University

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - views
 - Stored Procedures
 - Functions

SQL File 2*

```

89 • ALTER TABLE Students DROP CONSTRAINT AgeCheck;
90
91 -- Question: How do you update the major for student with ID 1 to 'Data Science'?
92 • UPDATE Students SET Major = 'Data Science' WHERE StudentID = 1;
93
94 • SELECT * FROM Students;
95
96
97

```

Result Grid

StudentID	Name	Age	Major	Email
1	Aanchi	19	Data Science	aaaa@aa.com
2	Priyanka	21	Data Science	pppp@pp.com
3	Carlos	20	Computer Science	cccc@cc.com
4	Dixya	24	Mathematics	dd@dd.com
5	Elena	18	Physics	ee@ee.com
6	Fahran	22	Computer Science	ff@ff.com
7	Grace	23	Unknown	gg@gg.com
8	Hannah	20	Computer Science	hh@hh.com
9	Irene	21	Mathematics	ii@ii.com
10	Jayden	24	Computer Science	jj@jj.com
11	Kara	22	Physics	kk@kk.com
12	Sharmu	19	Data Science	ss@ss.com
13	Maya	23	Computer Science	mm@mm.com
14	Nel	17	Undeclared	nn@nn.com

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

MySQL Workbench - University

SQL File 7:

```

86 • ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17);
87
88 -- Question: How do you remove the age-checking rule?
89 • ALTER TABLE Students DROP CONSTRAINT AgeCheck;
90
91 -- Question: How do you update the major for student with ID 1 to 'Data Science'?
92 • UPDATE Students SET Major = 'Data Science' WHERE StudentID = 1;
93
94 • SELECT * FROM Students;
95
96 -- Question: How do you remove all student records for students younger than 18?
97 • DELETE FROM Students
98 WHERE StudentID IN (
99     SELECT s.StudentID
100    FROM (
101        SELECT StudentID FROM Students WHERE Age < 18
102    ) AS s
103 );
104
105 • SELECT * FROM Students;
106
107
108
109
110
111

```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	17:31:29	DELETE FROM Students WHERE Age < 18	Error Code: 1175. You are using safe update mode and you tried to update a table without a WHERE that uses a key column in the WHERE or UPDATE clause.	0.000 sec / 0.000 sec
2	17:34:32	DELETE FROM Students WHERE StudentID IN (SELECT s.StudentID FROM (SELECT StudentID F...	1 row(s) affected	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench - University

SQL File 8:

```

101 • SELECT StudentID FROM Students WHERE Age < 18
102
103 ) AS s
104
105 • SELECT * FROM Students;

```

Result Grid

StudentID	Name	Age	Major	Email
1	Aarohi	19	Data Science	aarohi@example.com
2	Praaneeth	21	Data Science	praneeth@example.com
3	Carlos	20	Computer Science	carlos@example.com
4	Divya	24	Mathematics	divya@example.com
5	Elena	18	Physics	elena@example.com
6	Fahran	22	Computer Science	fahran@example.com
7	Grace	23	Undeclared	grace@example.com
8	Haley	20	Computer Science	haley@example.com
9	Irene	21	Mathematics	irene@example.com
10	Jayden	24	Computer Science	jayden@example.com
11	Kara	22	Physics	kara@example.com
12	Shannu	19	Data Science	shannu@example.com
13	Maya	23	Computer Science	may@example.com
14	Omar	20	Mathematics	omar@example.com
15	Omar	20	Mathematics	omar@example.com

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	17:36:45	SELECT * FROM Students LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench - University

SQL File 9:

```

107 • Question: Show the name and major of all students who are older than 19.
108 • SELECT Name, Major FROM Students WHERE Age > 19;
109
110
111

```

Result Grid

Name	Major
Grace	Data Science
Haley	Computer Science
Divya	Mathematics
Fahran	Computer Science
Grace	Undeclared
Haley	Computer Science
Irene	Mathematics
Jayden	Computer Science
Kara	Physics
Maya	Computer Science
Omar	Mathematics

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	17:36:45	SELECT * FROM Students LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
2	17:36:04	SELECT Name, Major FROM Students WHERE Age > 19 LIMIT 0, 1000	11 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- university
 - Tables
 - Views
 - Stored Procedures
 - Functions

SQL File 2*

```

107 -- Question: Show the name and major of all students who are older than 19.
108 • SELECT Name, Major FROM Students WHERE Age > 19;
109
110 -- Question: What is the average age of all students?
111 • SELECT AVG(Age) AS AvgAge FROM Students;
  
```

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid

Result 9 x

Action Output

#	Time	Action	Message	Duration / Fetch
1	17.36.45	SELECT * FROM Students LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
2	17.38.04	SELECT Name, Major FROM Students WHERE Age > 19 LIMIT 0, 1000	11 row(s) returned	0.000 sec / 0.000 sec
3	17.39.55	SELECT AVG(Age) AS AvgAge FROM Students LIMIT 0, 1000	1 row(s) returned	0.015 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- university
 - Tables
 - Views
 - Stored Procedures
 - Functions

SQL File 2*

```

111 • SELECT AVG(Age) AS AvgAge FROM Students;
112
113 -- Question: Which majors have more than 4 students, and what is the count for each?
114 • SELECT Major, COUNT(*) AS StudentCount
115   FROM Students
116   GROUP BY Major
117   HAVING COUNT(*) > 4;
118
119
120
121
  
```

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid

Result 11 x

Action Output

#	Time	Action	Message	Duration / Fetch
1	17.36.45	SELECT * FROM Students LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
2	17.38.04	SELECT Name, Major FROM Students WHERE Age > 19 LIMIT 0, 1000	11 row(s) returned	0.000 sec / 0.000 sec
3	17.39.55	SELECT AVG(Age) AS AvgAge FROM Students LIMIT 0, 1000	1 row(s) returned	0.015 sec / 0.000 sec
4	17.41.20	SELECT Major, COUNT(*) AS StudentCount FROM Students GROUP BY Major HAVING COUNT(*) > 5 LIMIT 0, 1000	0 row(s) returned	0.016 sec / 0.000 sec
5	17.41.35	SELECT Major, COUNT(*) AS StudentCount FROM Students GROUP BY Major HAVING COUNT(*) > 4 LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- university
 - Tables
 - Views
 - Stored Procedures
 - Functions

SQL File 2*

```

111 • SELECT AVG(Age) AS AvgAge FROM Students;
112
113 -- Question: Which majors have more than 4 students, and what is the count for each?
114 • SELECT Major, COUNT(*) AS StudentCount
115   FROM Students
116   GROUP BY Major
117   HAVING COUNT(*) > 4;
118
119 -- Question: List all information for students older than 20 and in 'Computer Science'.
120 • SELECT * FROM Students WHERE Age > 20 AND Major = 'Computer Science';
121
  
```

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid

Students 12 x

Action Output

StudentID	Name	Age	Major	Email
6	Fatima	22	Computer Science	NULL
10	Syden	24	Computer Science	NULL
13	Maya	23	Computer Science	NULL
14	NULL	NULL	NULL	NULL

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - views
 - Stored Procedures
 - Functions

SQL File 2

```

113 -- Question: Which majors have more than 4 students, and what is the count for each?
114 • SELECT Major, COUNT(*) AS StudentCount
115 FROM Students
116 GROUP BY Major
117 HAVING COUNT(*) > 4;
118
119 -- Question: List all information for students older than 20 and in 'Computer Science'.
120 • SELECT * FROM Students WHERE Age > 20 AND Major = 'Computer Science';
121
122 -- Question: How can you rank students based on their grades?
123 • SELECT s.Name, e.Grade,

```

Result Grid

Name	Grade	RankInClass
Aarohi	A	1
Carroll	A	1
Irene	A	1
Farhan	A-	4
Aarohi	B	5
Pranayanthi	B	5
Pranayanthi	B	5
Grace	B+	8
Divya	C	9
Harsh	C+	10

Administration Schemas Information

Schema: university

Result 13

Action Output

#	Time	Action	Message	Duration / Fetch
3	17:39:55	SELECT AVG(Age) AS AvgAge FROM Students LIMIT 0, 1000	1 row(s) returned	0.015 sec / 0.000 sec
4	17:41:20	SELECT Major, COUNT(*) AS StudentCount FROM Students GROUP BY Major HAVING COUNT(*) > 5 LIMIT... 0 row(s) returned		0.016 sec / 0.000 sec
5	17:41:35	SELECT Major, COUNT(*) AS StudentCount FROM Students GROUP BY Major HAVING COUNT(*) > 4 LIMIT... 1 row(s) returned		0.000 sec / 0.000 sec
6	17:45:39	SELECT * FROM Students WHERE Age > 20 AND Major = 'Computer Science' LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec
7	17:47:10	SELECT s.Name, e.Grade, RANK() OVER (ORDER BY Grade ASC) AS RankInClass FROM Enrollments e JOIN Students s ON e.StudentID = s.StudentID	10 row(s) returned	0.016 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - views
 - Stored Procedures
 - Functions

SQL File 2

```

123 • SELECT s.Name, e.Grade,
124 RANK() OVER (ORDER BY Grade ASC) AS RankInClass
125 FROM Enrollments e JOIN Students s ON e.StudentID = s.StudentID;
126
127 -- Question: List student names and the courses they are enrolled in.
128 • SELECT s.Name, c.CourseName
129 FROM Students s
130 INNER JOIN Enrollments e ON s.StudentID = e.StudentID;
131 INNER JOIN Courses c ON e.CourseID = c.CourseID;
132

```

Result Grid

Name	CourseName
Aarohi	Database Systems
Pranayanthi	Machine Learning
Carlos	Algorithms
Divya	Machine Learning
Farhan	Database Systems
Grace	Machine Learning
Harsh	Algorithms
Irene	Machine Learning
Pranayanthi	Database Systems

Administration Schemas Information

Schema: university

Result 14

Action Output

#	Time	Action	Message	Duration / Fetch
3	17:39:55	SELECT AVG(Age) AS AvgAge FROM Students LIMIT 0, 1000	1 row(s) returned	0.015 sec / 0.000 sec
4	17:41:20	SELECT Major, COUNT(*) AS StudentCount FROM Students GROUP BY Major HAVING COUNT(*) > 5 LIMIT... 0 row(s) returned		0.016 sec / 0.000 sec
5	17:41:35	SELECT Major, COUNT(*) AS StudentCount FROM Students GROUP BY Major HAVING COUNT(*) > 4 LIMIT... 1 row(s) returned		0.000 sec / 0.000 sec
6	17:45:39	SELECT * FROM Students WHERE Age > 20 AND Major = 'Computer Science' LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec
7	17:47:10	SELECT s.Name, e.Grade, RANK() OVER (ORDER BY Grade ASC) AS RankInClass FROM Enrollments e JOIN Students s ON e.StudentID = s.StudentID	10 row(s) returned	0.016 sec / 0.000 sec
8	17:48:31	SELECT s.Name, c.CourseName FROM Students s INNER JOIN Enrollments e ON s.StudentID = e.StudentID	10 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - views
 - Stored Procedures
 - Functions

SQL File 2

```

133 -- Question: Show all students and their courses, including students not enrolled in any course.
134 • SELECT s.Name, c.CourseName
135 FROM Students s
136 LEFT JOIN Enrollments e ON s.StudentID = e.StudentID
137 LEFT JOIN Courses c ON e.CourseID = c.CourseID;
138
139

```

Result Grid

Name	CourseName
Aarohi	Database Systems
Pranayanthi	Machine Learning
Pranayanthi	Database Systems
Carlos	Algorithms
Divya	Machine Learning
Elena	
Farhan	Database Systems
Grace	Machine Learning
Harsh	Algorithms
Irene	Machine Learning
Jayden	
Kara	
Shannu	
Mari	
Omar	

Administration Schemas Information

Schema: university

Result 15

Action Output

#	Time	Action	Message	Duration / Fetch
5	17:41:35	SELECT Major, COUNT(*) AS StudentCount FROM Students GROUP BY Major HAVING COUNT(*) > 4 LIMIT... 1 row(s) returned		0.000 sec / 0.000 sec
6	17:45:39	SELECT * FROM Students WHERE Age > 20 AND Major = 'Computer Science' LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec
7	17:47:10	SELECT s.Name, e.Grade, RANK() OVER (ORDER BY Grade ASC) AS RankInClass FROM Enrollments e JOIN Students s ON e.StudentID = s.StudentID	10 row(s) returned	0.016 sec / 0.000 sec
8	17:48:31	SELECT s.Name, c.CourseName FROM Students s INNER JOIN Enrollments e ON s.StudentID = e.StudentID	10 row(s) returned	0.000 sec / 0.000 sec
9	17:50:10	SELECT s.Name, c.CourseName FROM Students s LEFT JOIN Enrollments e ON s.StudentID = e.StudentID	16 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'university' schema with its tables: students, courses, and enrollments. The main area contains a query editor with the following SQL code:

```
138
139 -- Question: Generate every possible combination of a student paired with a course.
140 SELECT s.Name, c.CourseName
141 FROM Students s CROSS JOIN Courses c;
142
143
```

The results grid shows the output of the query:

Name	CourseName
Aarohi	Machine Learning
Aarohi	Algorithms
Aarohi	Database Systems
Praneeth	Web Development
Praneeth	Machine Learning
Praneeth	Algorithms
Praneeth	Database Systems
Carlos	Web Development
Carlos	Machine Learning
Carlos	Algorithms
Carlos	Database Systems
Divya	Web Development
Divya	Machine Learning
Divya	Algorithms
Divya	Database Systems
Bena	Web Development
Bena	Machine Learning
Bena	Algorithms
Bena	Database Systems
Fahran	Web Development
Fahran	Machine Learning

The right sidebar includes tabs for Result Grid, Filter Rows, Export, Wrap Cell Content, and various tools like Result Set, Form Editor, Field Types, Query Editor, and Execution Plan.

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

- File Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Schemas:** University is selected.
- SQL Editor:** Contains the following SQL code:


```

141 FROM Students s CROSS JOIN Courses c;
142
143 -- Question: Find pairs of different students who are in the same major.
144 • SELECT s1.Name AS Student1, s2.Name AS Student2
145 FROM Students s1
146 JOIN Students s2 ON s1.Major = s2.Major AND s1.StudentID <> s2.StudentID
      
```
- Result Grid:** Displays the results of the query, showing pairs of students with the same major:

Student1	Student2
Shanu	Aarohi
Proneth	Aarohi
Shanu	Proneth
Aarohi	Proneth
Maya	Carlos
Jayden	Carlos
Harsh	Carlos
Farhan	Carlos
Omar	Driya
Irene	Driya
Kara	Elena
Maya	Farhan
Jayden	Farhan
Harsh	Farhan
Farhan	Harsh
Carlos	Harsh
Omar	Irene
Nimra	Irene
- Output Tab:** Action Output section shows the execution of the query with the following log entries:

#	Action	Time	Message	Duration / Fetch
1	Action Output			
2	SELECT s.Name, c.CourseName FROM Students s INNER JOIN Enrollments e ON s.StudentID = e.StudentID ... 10 rows(s) returned	8 17:50:31		0.000 sec / 0.000 sec
3	SELECT s.Name, c.CourseName FROM Students s LEFT JOIN Enrollments e ON s.StudentID = e.StudentID ... 16 rows(s) returned	9 17:50:31		0.000 sec / 0.000 sec
4	SELECT s1.Name AS Student1, s2.Name AS Student2	10 17:50:03		66.666 ms returned
5	FROM Students s1			0.000 sec / 0.000 sec
6	JOIN Students s2 ON s1.Major = s2.Major AND s1.StudentID <> s2.StudentID			0.000 sec / 0.000 sec

The screenshot shows the MySQL Workbench interface with the 'university' schema selected. In the SQL Editor, a query is run:

```
146 JOIN Students s2 ON s1.Major = s2.Major AND s1.StudentID <> s2.StudentID
147
148 -- Question: For each major, provide a comma-separated list of student names.
149 • SELECT Major,
150   GROUP_CONCAT(`Name` ORDER BY `Name` SEPARATOR ', ') AS Students
151 FROM Students
152 GROUP BY Major
153
```

The results are displayed in the Result Grid:

Major	Students
Computer Science	Carlos, Farhan, Harsh, Jayden, Maya
Data Sciences	Aarohi, Praneeth, Shannu
Mathematics	Divya, Irene, Omar
Physics	Ella, Kara
Undeclared	Grace

The status bar at the bottom right indicates the duration of the query execution.

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, Help. The left sidebar has a Navigator section showing SCHEMAS (university) and TABLES (courses, assignments, students). The main area displays a query editor with the following code:

```
-- Question: Find students who are older than the average student age.  
SELECT Name FROM Students  
WHERE Age > (SELECT AVG(Age) FROM Students);
```

The results grid shows the names of students older than the average age:

Name
Divya
Farhan
Gauri
Jayden
Kara
Maya

A context help panel on the right side provides information about automatic context help being disabled, with a link to the toolbar manual for help.

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

- File Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Schemas:** university
- Tables:** courses, enrolments, students
- SQL Editor:** Contains a multi-line SQL script:

```
157 WHERE Age > (SELECT AVG(Age) FROM Students);  
158  
159 -- Question: Get the names of students who have received a grade 'A' in any course.  
160 • SELECT Name FROM Students s;  
161 ◊ WHERE EXISTS (  
162     SELECT 1 FROM Enrolments e  
163     WHERE e.StudentID = s.StudentID AND e.Grade = 'A'  
164 );  
165
```
- Result Grid:** Shows the names of students: Aardi, Carlos, Irene.
- Timeline:** Shows the execution of three queries:

#	Action	Time	Message	Duration / Fetch
1	SELECT Major, GROUP_CONCAT(Name ORDER BY Name SEPARATOR ',') AS Students FROM Student;	17:59:10	5 row(s) returned	0.000 sec / 0.000 sec
2	SELECT Name FROM Students WHERE Age > (SELECT AVG(Age) FROM Students)	18:01:02	6 row(s) returned	0.000 sec / 0.000 sec
3	SELECT Name FROM Students s WHERE EXISTS (SELECT 1 FROM Enrolments e WHERE e.StudentID = s.StudentID)	18:02:10	3 row(s) returned	0.016 sec / 0.000 sec
- Help:** Automatic context help is enabled.

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

- File**: University
- Toolbars**: Standard MySQL Workbench toolbar.
- Schemas**: University schema selected.
- SQL Editor (SQL File 2*)**: Contains the following SQL code:

```
166 -- Question: Display each major and its calculated average age.
167 • SELECT Major, ROUND(AvgAge) AS AvgAge
168   FROM (
169     SELECT Major, AVG(Age) AS AvgAge
170       FROM Students
171      GROUP BY Major
172    ) AS t;
```
- Result Grid**: Displays the results of the query:

Major	AvgAge
Data Science	20
Computer Science	22
Mathematics	22
Physics	20
Undeclared	23
- Toolbars**: Standard MySQL Workbench toolbar.
- Right Panel**: Shows the following sections:
 - SQLAdditions
 - Automatic context help disabled. Use the toolbar or manually get help for the current caret position or toggle automatic help.
 - Result Grid
 - Form Editor
 - Field Types
 - Query Stats
- Bottom Bar**: Read Only, Context Help, SkipPrev.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS sys university Tables courses enrolments students Views Stored Procedures Functions

SQL File 2

```

171   GROUP BY Major
172 ) AS t;
173
174 -- Question: Create a single list of all student names and course names.
175 • SELECT Name FROM Students
176 UNION
177 SELECT CourseName FROM Courses

```

Result Grid

Name
Aarohi
Himasheth
Carlo
Divya
Elena
Farhan
Grace
Harsh
Irene
Jayden
Kara
Shanno
Maya
Omar
Danhieu...
Algorithms...
Machine...
Web De...

Administration Schemas Information Schema: university

Result 23

Action Output

#	Time	Action	Message	Duration / Fetch
4	18:03:21	SELECT Major, AvgAge FROM (SELECT Major, AVG(Age) AS AvgAge FROM Students GROUP BY Major) AS t;	5 row(s) returned	0.000 sec / 0.000 sec
5	18:04:54	SELECT Major, ROUND(AvgAge) AS AvgAge FROM (SELECT Major, AVG(Age) AS AvgAge FROM Students) AS t;	5 row(s) returned	0.000 sec / 0.000 sec
6	18:06:47	SELECT Name FROM Students UNION SELECT CourseName FROM Courses	18 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS sys university Tables courses enrolments students Views Stored Procedures Functions

SQL File 2

```

173
174 -- Question: Create a single list of all student names and course names.
175 • SELECT Name FROM Students
176 UNION
177 SELECT CourseName FROM Courses
178
179 -- Question: Find student IDs present in both Students and Enrollments tables.
180 • SELECT StudentID FROM Students
181 INTERSECT
182 • SELECT StudentID FROM Enrollments;
183
184
185
186

```

Result Grid

StudentID
1
2
3
4
5
6
7
8
9

Administration Schemas Information Schema: university

Result 24

Action Output

#	Time	Action	Message	Duration / Fetch
5	18:04:54	SELECT Major, ROUND(AvgAge) AS AvgAge FROM (SELECT Major, AVG(Age) AS AvgAge FROM Students) AS t;	5 row(s) returned	0.000 sec / 0.000 sec
6	18:06:47	SELECT Name FROM Students UNION SELECT CourseName FROM Courses	18 row(s) returned	0.000 sec / 0.000 sec
7	18:08:44	SELECT StudentID FROM Students INTERSECT SELECT StudentID FROM Enrollments	8 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS sys university Tables courses enrolments students Views Stored Procedures Functions

SQL File 2

```

183
184 -- Question: Find students registered but not enrolled in any course.
185 • SELECT Name
186   FROM Students
187   WHERE StudentID NOT IN (
188     SELECT StudentID FROM Enrollments
189   );
190

```

Result Grid

Name
Elena
Jayden
Kara
Shanno
Maya
Omar

Administration Schemas Information Schema: university

Students 26

Action Output

#	Time	Action	Message	Duration / Fetch
7	18:08:44	SELECT StudentID FROM Students INTERSECT SELECT StudentID FROM Enrollments	8 row(s) returned	0.000 sec / 0.000 sec
8	18:10:58	SELECT Name FROM Students WHERE StudentID NOT IN (SELECT StudentID FROM Enrollments) EXCEPT	6 row(s) returned	0.000 sec / 0.000 sec
9	18:12:58	SELECT Name FROM Students WHERE StudentID NOT IN (SELECT StudentID FROM Enrollments)	6 row(s) returned	0.015 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

Dont Limit

```

185 • SELECT Name
186   FROM Students
187   WHERE StudentID NOT IN (
188     SELECT StudentID FROM Enrollments
189   )
190
191 -- Question: How to perform several changes as a single atomic operation?
192 • START TRANSACTION;
193 • INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade)
194   VALUES (16, 1, 104, 'A');
195 • UPDATE Students
196   SET Major = 'AI'
197   WHERE StudentID = 1;
198 • COMMIT;
199
200
201
202
203

```

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

Administration Schemas Information

Schema: university

Action Output

#	Time	Action	Message	Duration / Fetch
5	18:04:54	SELECT Major,ROUND(AvgAge) AS AvgAge FROM (SELECT Major, AVG(Age) AS AvgAge FROM Students) t GROUP BY Major	5 row(s) returned	0.000 sec / 0.000 sec
6	18:06:47	SELECT Name FROM Students UNION SELECT CourseName FROM Courses	18 row(s) returned	0.000 sec / 0.000 sec
7	18:08:44	SELECT StudentID FROM Students INTERSECT SELECT StudentID FROM Enrollments	5 row(s) returned	0.000 sec / 0.000 sec
8	18:10:08	SELECT Name FROM Students WHERE StudentID IN (SELECT StudentID FROM Students EXCEPT SELECT StudentID FROM Enrollments)	5 row(s) returned	0.000 sec / 0.000 sec
9	18:12:58	SELECT Name FROM Students WHERE StudentID NOT IN (SELECT StudentID FROM Enrollments)	5 row(s) returned	0.015 sec / 0.000 sec
10	18:15:36	START TRANSACTION	0 row(s) affected	0.000 sec
11	18:15:36	INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade) VALUES (16, 1, 104, 'A')	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
12	18:15:36	UPDATE Students SET Major = 'AI' WHERE StudentID = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
13	18:15:36	COMMIT;	0 row(s) affected	0.016 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

Dont Limit

```

185 • SELECT Name
186   FROM Students
187   WHERE StudentID NOT IN (
188     SELECT StudentID FROM Enrollments
189   )
190
191 -- Question: How to perform several changes as a single atomic operation?
192 • START TRANSACTION;
193 • INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade)
194   VALUES (16, 1, 104, 'A');
195 • UPDATE Students
196   SET Major = 'AI'
197   WHERE StudentID = 1;
198 • COMMIT;
199
200 -- Question: How do you create an index to speed up searches on the Major column?
201 • CREATE INDEX idx_student_major ON Students(Major);
202
203
204

```

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

Administration Schemas Information

Schema: university

Action Output

#	Time	Action	Message	Duration / Fetch
6	18:04:47	SELECT Name FROM Students UNION SELECT CourseName FROM Courses	18 row(s) returned	0.000 sec / 0.000 sec
7	18:06:44	SELECT StudentID FROM Students INTERSECT SELECT StudentID FROM Enrollments	0 row(s) returned	0.000 sec / 0.000 sec
8	18:08:44	SELECT Name FROM Students WHERE StudentID IN (SELECT StudentID FROM Students EXCEPT SELECT StudentID FROM Enrollments)	0 row(s) returned	0.000 sec / 0.000 sec
9	18:12:58	SELECT Name FROM Students WHERE StudentID NOT IN (SELECT StudentID FROM Enrollments)	0 row(s) returned	0.015 sec / 0.000 sec
10	18:15:36	START TRANSACTION	0 row(s) affected	0.000 sec
11	18:15:36	INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade) VALUES (16, 1, 104, 'A')	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
12	18:15:36	UPDATE Students SET Major = 'AI' WHERE StudentID = 1	0 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
13	18:15:36	COMMIT;	0 row(s) affected	0.016 sec
14	18:16:40	CREATE INDEX idx_student_major ON Students(Major)	0 rows affected Records: 0 Duplicates: 0 Warnings: 0	0.062 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- sys
- university
 - Tables
 - courses
 - enrollments
 - students
 - Views
 - Stored Procedures
 - Functions

Dont Limit

```

190 • COMMIT;
191
192 -- Question: How do you create an index to speed up searches on the Major column?
193 • CREATE INDEX idx_student_major ON Students(Major);
194
195 -- Question: How can you check if the index is used?
196 • SELECT * FROM Students WHERE Major = 'Data Science';
197

```

SQL File 2

Result Grid

StudentID	Name	Age	Major	Email
2	Praneth	21	Data Science	praneth@gmail.com
12	Shannu	19	Data Science	shannu@gmail.com

Form Editor

Field Types

Students 27

Output

Action Output

#	Time	Action	Message	Duration / Fetch
7	18:08:44	SELECT StudentID FROM Students INTERSECT SELECT StudentID FROM Enrollments	8 row(s) returned	0.000 sec / 0.000 sec
8	18:10:08	SELECT Name FROM Students WHERE StudentID IN (SELECT StudentID FROM Students EXCEPT SELECT StudentID FROM Enrollments)	5 row(s) returned	0.000 sec / 0.000 sec
9	18:12:58	SELECT Name FROM Students WHERE StudentID NOT IN (SELECT StudentID FROM Enrollments)	5 row(s) returned	0.015 sec / 0.000 sec
10	18:15:36	START TRANSACTION	0 row(s) affected	0.000 sec
11	18:15:36	INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade) VALUES (16, 1, 104, 'A')	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
12	18:15:36	UPDATE Students SET Major = 'AI' WHERE StudentID = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
13	18:15:36	COMMIT;	0 row(s) affected	0.016 sec
14	18:16:40	CREATE INDEX idx_student_major ON Students(Major)	0 rows affected Records: 0 Duplicates: 0 Warnings: 0	0.062 sec
15	18:17:53	SELECT * FROM Students WHERE Major = 'Data Science'	2 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS Filter objects sys university courses enrolments students Views Stored Procedures Functions

SQL File 2 * DONT LIMIT

```

194 VALUES (16, 1, 104, 'A');
195 UPDATE Students
196 SET Major = 'AI'
197 WHERE StudentID = 1;
198 COMMIT;
199
200 -- Question: How do you create an index to speed up searches on the Major column?
201 CREATE INDEX idx_student_major ON Students(Major);
202
203 -- Question: How can you check if the index is used?
204 SELECT * FROM Students WHERE Major = 'Data Science';
205
206 -- Question: How do you remove the index?
207 DROP INDEX idx_student_major ON Students;
208
209
210
211
212

```

Output:

Action Output

Time	Action	Message	Duration / Fetch
8 18:10:08	SELECT Name FROM Students WHERE StudentID IN (SELECT StudentID FROM Students EXCEPT ...)	0 row(s) returned	0.000 sec / 0.000 sec
9 18:12:58	SELECT Name FROM Students WHERE StudentID NOT IN (SELECT StudentID FROM Enrolments)	6 row(s) returned	0.015 sec / 0.000 sec
10 18:15:36	START TRANSACTION	0 row(s) affected	0.000 sec
11 18:15:36	INSERT INTO Enrolments (EnrollmentID, StudentID, CourseID, Grade) VALUES (16, 1, 104, 'A')	1 row(s) affected	0.000 sec
12 18:15:36	UPDATE Students SET Major = 'AI' WHERE StudentID = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
13 18:15:36	COMMIT	0 row(s) affected	0.016 sec
14 18:16:40	CREATE INDEX idx_student_major ON Students(Major)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.062 sec
15 18:17:53	SELECT * FROM Students WHERE Major = 'Data Science'	2 row(s) returned	0.000 sec / 0.000 sec
16 18:19:17	DROP INDEX idx_student_major ON Students	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec

Schema: university

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS Filter objects sys university tables courses enrolments students Views Stored Procedures Functions

SQL File 2 * DONT LIMIT

```

207 DROP INDEX idx_student_major ON Students;
208
209 -- Question: List all students, sorted by age in descending order.
210 SELECT Name, Age
211 FROM Students
212 ORDER BY Age DESC;
213
214

```

Result Grid

Name	Age
Divya	24
Jayde	24
Grace	23
Mary	23
Farhan	22
Kiera	22
Praneth	21
Irene	21
Carlos	20
Harsh	20
Omar	20
Aarohi	19
Shannu	19
Elena	18

Students 28 x

Output:

Action Output

Time	Action	Message	Duration / Fetch
14 18:16:40	CREATE INDEX idx_student_major ON Students(Major)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.062 sec
15 18:17:53	SELECT * FROM Students WHERE Major = 'Data Science'	2 row(s) returned	0.000 sec / 0.000 sec
16 18:19:17	DROP INDEX idx_student_major ON Students	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec
17 18:20:06	SELECT Name, Age FROM Students ORDER BY Age DESC	14 row(s) returned	0.000 sec / 0.000 sec

Schema: university

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS Filter objects sys university tables courses enrolments students Views Stored Procedures Functions

SQL File 2 * DONT LIMIT

```

211 FROM Students
212 ORDER BY Age DESC;
213
214 -- Question: List students, sorted first by age (descending), then by name (ascending).
215 SELECT Name, Age
216 FROM Students
217 ORDER BY Age DESC, Name ASC;
218

```

Result Grid

Name	Age
Divya	24
Jayde	24
Grace	23
Mary	23
Farhan	22
Kiera	22
Irene	21
Praneth	21
Carlos	20
Harsh	20
Omar	20
Aarohi	19
Shannu	19
Elena	18

Students 29 x

Output:

Action Output

Time	Action	Message	Duration / Fetch
15 18:17:53	SELECT * FROM Students WHERE Major = 'Data Science'	2 row(s) returned	0.000 sec / 0.000 sec
16 18:19:17	DROP INDEX idx_student_major ON Students	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec
17 18:20:06	SELECT Name, Age FROM Students ORDER BY Age DESC	14 row(s) returned	0.000 sec / 0.000 sec
18 18:21:13	SELECT Name, Age FROM Students ORDER BY Age DESC, Name ASC	14 row(s) returned	0.000 sec / 0.000 sec

Schema: university

Object Info Session

MySQL Workbench

University

File Edit View Query Database Server Tools Scripting Help

Navigator: Schemas

SCHEMAS: university

Tables: courses, enrollments, students

Views, Stored Procedures, Functions

SQL File 2: x

```

217 ORDER BY Age DESC, Name ASC;
218
-- Question: Show the top 5 oldest students.
219
220 • SELECT Name, Age
221   FROM Students
222   ORDER BY Age DESC
223   LIMIT 5;
224

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

Name	Age
Dawn	24
Jayden	24
Grace	23
Maya	23
Farhan	22

Administration Schemas

Schema: university

Students 30 x

Action Output

#	Time	Action	Message	Duration / Fetch
16	18:19:17	DROP INDEX dx_student_major ON Students	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec
17	18:20:06	SELECT Name, Age FROM Students ORDER BY Age DESC	14 row(s) returned	0.000 sec / 0.000 sec
18	18:21:13	SELECT Name, Age FROM Students ORDER BY Age DESC, Name ASC	14 row(s) returned	0.000 sec / 0.000 sec
19	18:20:06	SELECT Name, Age FROM Students ORDER BY Age DESC LIMIT 5	5 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench

University

File Edit View Query Database Server Tools Scripting Help

Navigator: Schemas

SCHEMAS: university

Tables: courses, enrollments, students

Views, Stored Procedures, Functions

SQL File 2: x

```

224
225
226 • WITH AvgAgeCTE AS (
227   SELECT AVG(Age) AS AgeValue FROM Students
228 )
229
230   SELECT Name, Age
231   FROM Students, AvgAgeCTE
232   WHERE Students.Age > AvgAgeCTE.AgeValue;
233

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

Name	Age
Divya	24
Farhan	22
Grace	23
Jayden	24
Klara	22
Maya	23

Administration Schemas

Schema: university

Result 31 x

Action Output

#	Time	Action	Message	Duration / Fetch
17	18:20:06	SELECT Name, Age FROM Students ORDER BY Age DESC	14 row(s) returned	0.000 sec / 0.000 sec
18	18:21:13	SELECT Name, Age FROM Students ORDER BY Age DESC, Name ASC	14 row(s) returned	0.000 sec / 0.000 sec
19	18:20:06	SELECT Name, Age FROM Students ORDER BY Age DESC LIMIT 5	5 row(s) returned	0.000 sec / 0.000 sec
20	18:33:41	WITH AvgAgeCTE AS (SELECT AVG(Age) AS AgeValue FROM Students) SELECT Name, Age FROM St... 6 row(s) returned		0.016 sec / 0.000 sec

Object Info Session