

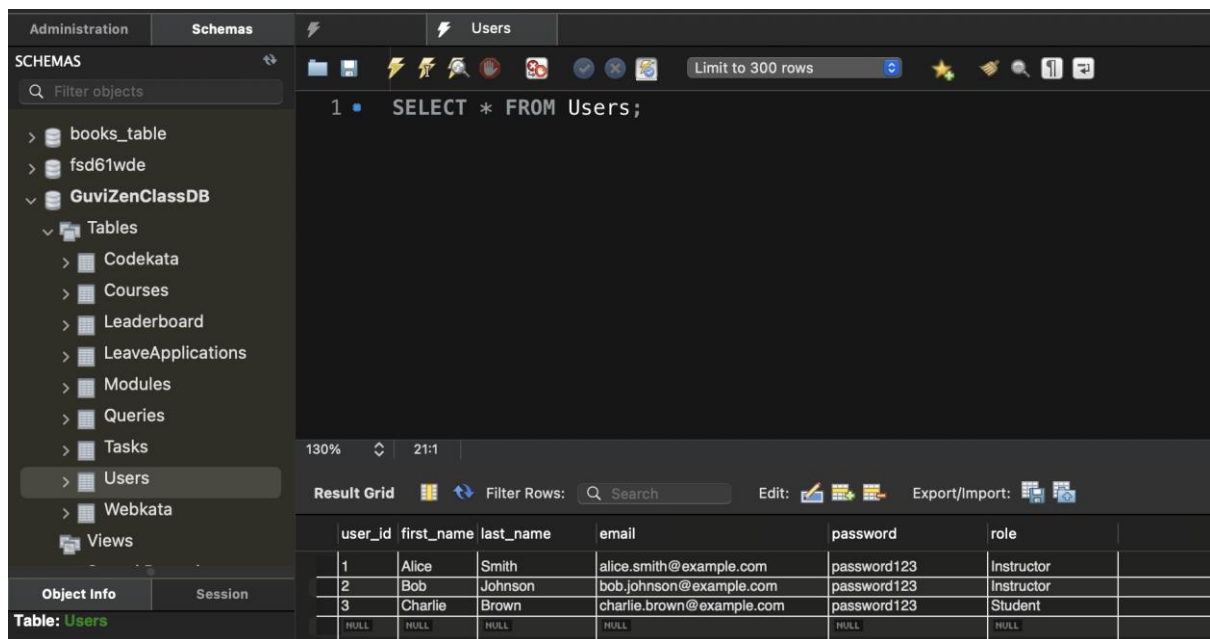
Design DB model for Guvi Zen class:

Users:

```
```sql
create table users (
 user_id int not null auto_increment,
 first_name varchar(50) not null,
 last_name varchar (50) not null,
 email varchar (100) not null unique,
 password varchar(255) not null,
 role varchar(20) not null,
 primary key (user_id)
);

insert into users (first_name, last_name, email, password, role)
values
('Alice', 'Smith', 'alice.smith@example.com', 'password123', 'Instructor'),
('Bob', 'Johnson', 'bob.johnson@example.com', 'password123', 'Instructor'),
('Charlie', 'Brown', 'charlie.brown@example.com', 'password123', 'Student');
```
```

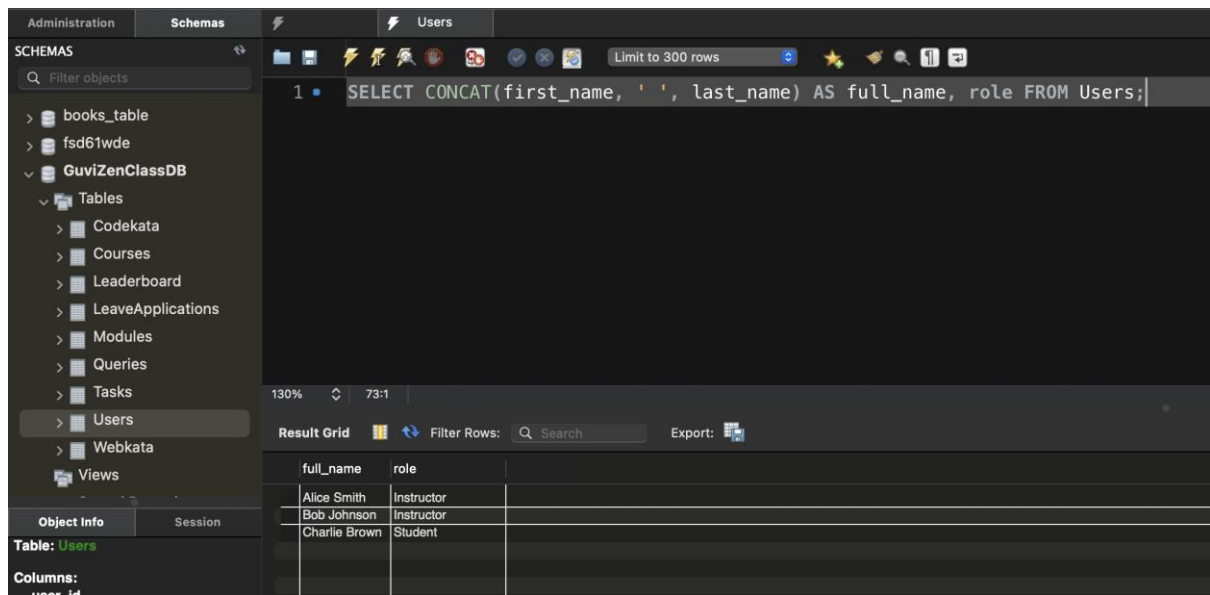
Output:



The screenshot displays a database management interface. On the left, a sidebar shows the 'GuviZenClassDB' schema with various tables listed. The 'Users' table is selected. The main area shows a query editor with the command 'SELECT * FROM Users;'. Below the query editor, a 'Result Grid' displays the data from the 'Users' table. The grid has columns for 'user_id', 'first_name', 'last_name', 'email', 'password', and 'role'. The data is as follows:

| user_id | first_name | last_name | email | password | role |
|---------|------------|-----------|---------------------------|-------------|------------|
| 1 | Alice | Smith | alice.smith@example.com | password123 | Instructor |
| 2 | Bob | Johnson | bob.johnson@example.com | password123 | Instructor |
| 3 | Charlie | Brown | charlie.brown@example.com | password123 | Student |

List all users, showing their full name and role from users table ?



Courses:

```
```sql
```

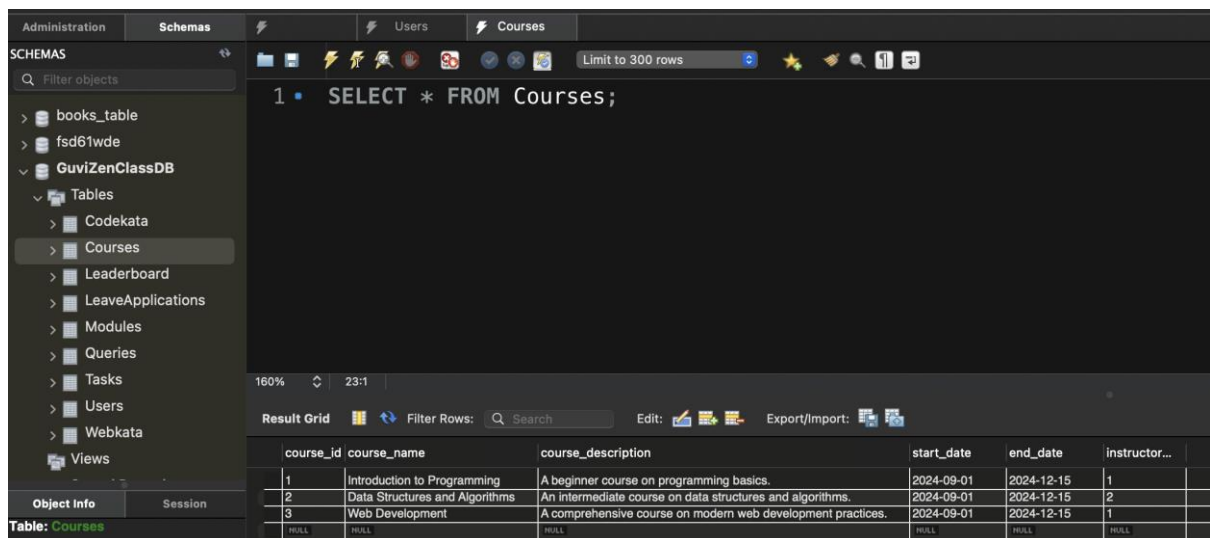
```
create table courses (
 course_id int not null auto_increment,
 course_name varchar(100) not null,
 course_description text,
 start_date date not null,
 end_date date not null,
 instructor_id int not null,
 primary key (course_id),
 foreign key (instructor_id) references users(user_id)
);
```

```
insert into courses (course_name, course_description, start_date, end_date,
instructor_id)
values
('Introduction to Programming', 'A beginner course on programming basics.',
'2024-09-01', '2024-12-15', 1),
('Data Structures and Algorithms', 'An intermediate course on data structures
and algorithms.', '2024-09-01', '2024-12-15', 2),
```

```
('Web Development', 'A comprehensive course on modern web development practices.', '2024-09-01', '2024-12-15', 1);

```

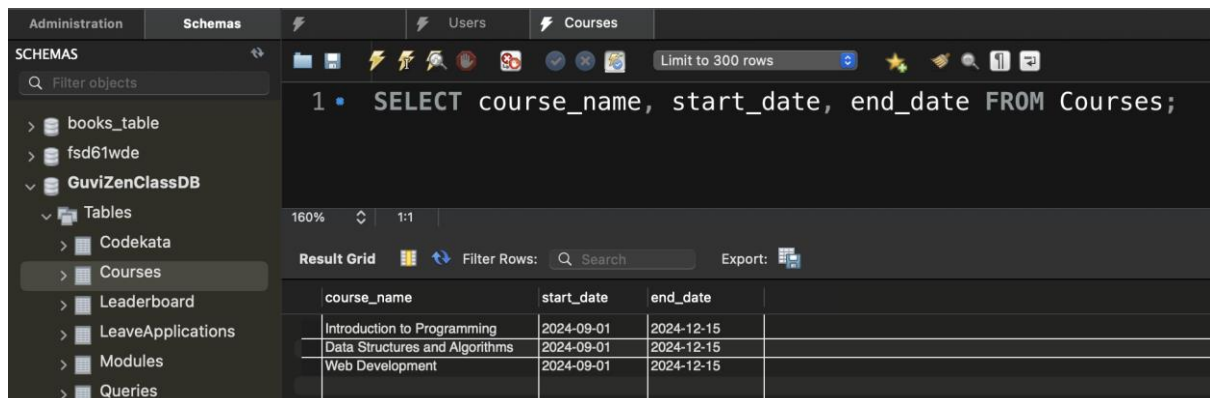
*Output:*



The screenshot shows a database management interface with a sidebar on the left containing a tree view of schemas and tables. The main area displays a SQL query: `1 • SELECT * FROM Courses;`. Below the query, a 'Result Grid' shows the data from the 'Courses' table. The table has columns: `course_id`, `course_name`, `course_description`, `start_date`, `end_date`, and `instructor...`. The data rows are:

course_id	course_name	course_description	start_date	end_date	instructor...
1	Introduction to Programming	A beginner course on programming basics.	2024-09-01	2024-12-15	1
2	Data Structures and Algorithms	An intermediate course on data structures and algorithms.	2024-09-01	2024-12-15	2
3	Web Development	A comprehensive course on modern web development practices.	2024-09-01	2024-12-15	1
NULL	NULL	NULL	NULL	NULL	NULL

*List all courses along with start and end date ?*



The screenshot shows the same database management interface, but the SQL query is now: `1 • SELECT course_name, start_date, end_date FROM Courses;`. The 'Result Grid' shows the data from the 'Courses' table, filtered by start and end date. The table has columns: `course_name`, `start_date`, and `end_date`. The data rows are:

course_name	start_date	end_date
Introduction to Programming	2024-09-01	2024-12-15
Data Structures and Algorithms	2024-09-01	2024-12-15
Web Development	2024-09-01	2024-12-15

**Modules:**

```
```sql
create table modules (
  module_id int not null auto_increment,
```

```

module_name varchar(100) not null,
module_description text,
course_id int not null,
primary key (module_id),
foreign key (course_id) references courses(course_id)
);

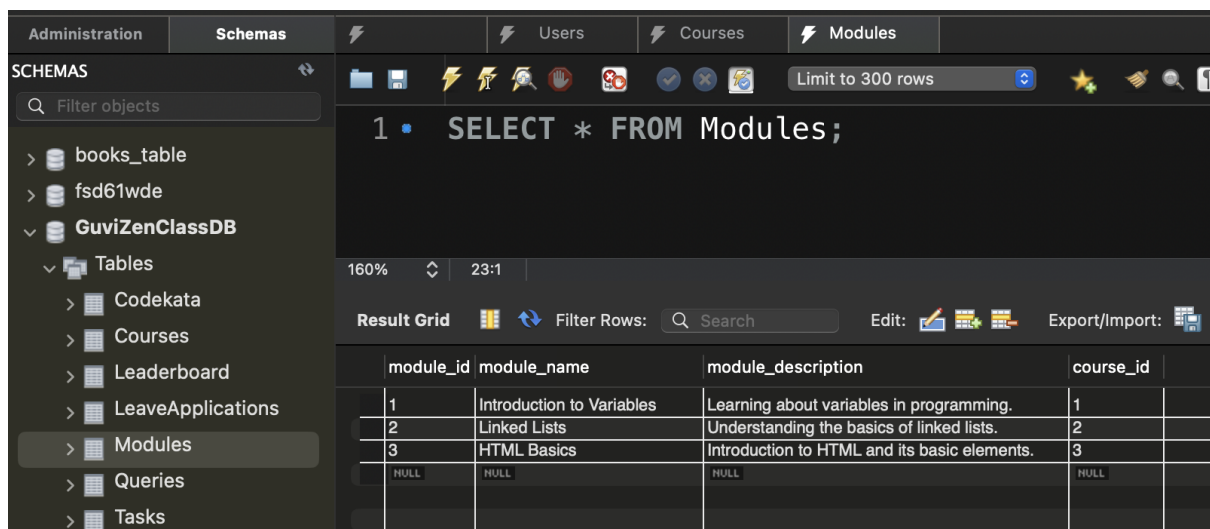
```

```

insert into modules (module_name, module_description, course_id)
values
('Introduction to Variables', 'Learning about variables in programming.', 1),
('Linked Lists', 'Understanding the basics of linked lists.', 2),
('HTML Basics', 'Introduction to HTML and its basic elements.', 3);
'''

```

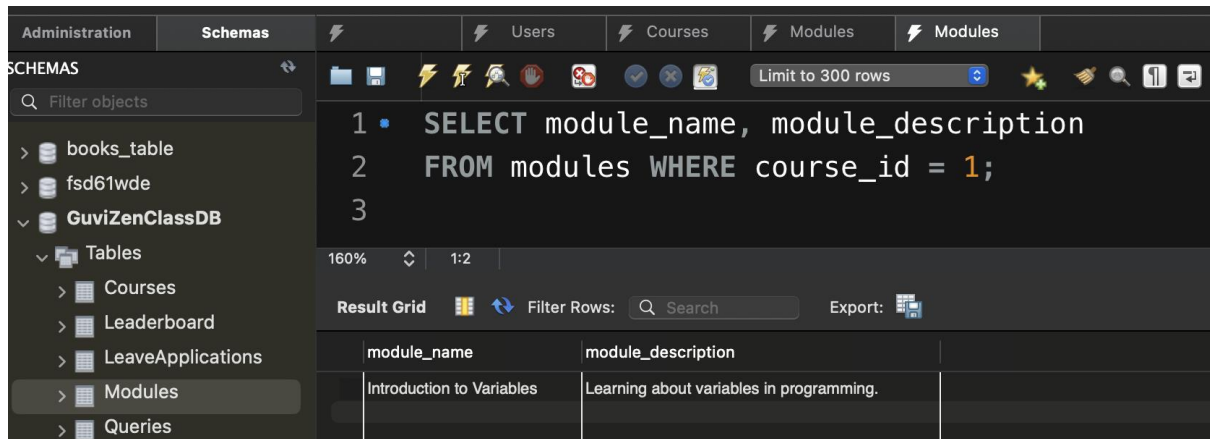
Output:



The screenshot shows a database management interface with a sidebar on the left containing a tree view of schemas. The 'GuvizZenClassDB' schema is expanded, showing tables like 'Codekata', 'Courses', 'Leaderboard', 'LeaveApplications', 'Modules', 'Queries', and 'Tasks'. The 'Modules' table is selected. The main area displays a SQL query: `SELECT * FROM Modules;` and a 'Result Grid' showing the data. The grid has columns for 'module_id', 'module_name', 'module_description', and 'course_id'. The data is as follows:

module_id	module_name	module_description	course_id
1	Introduction to Variables	Learning about variables in programming.	1
2	Linked Lists	Understanding the basics of linked lists.	2
3	HTML Basics	Introduction to HTML and its basic elements.	3
HULL	HULL	HULL	HULL

Module along with description for specific course identify by the course_id:



The screenshot shows a database management interface with a sidebar on the left displaying a schema tree. The main area contains a SQL query editor with the following query:

```
1 • SELECT module_name, module_description
2   FROM modules WHERE course_id = 1;
3
```

Below the query editor, there is a 'Result Grid' section showing the results of the query. The grid has two columns: 'module_name' and 'module_description'. The first row of data shows 'Introduction to Variables' and 'Learning about variables in programming.'.

module_name	module_description
Introduction to Variables	Learning about variables in programming.

Tasks:

```
```sql
create table tasks (
 task_id int not null auto_increment,
 module_id int not null,
 task_description text,
 due_date date,
 status varchar(20),
 primary key (task_id),
 foreign key (module_id) references modules(module_id)
);

insert into tasks (module_id, task_description, due_date, status) values
(1, 'complete the coding assignment on variables.', '2024-10-01', 'pending'),
(2, 'submit the linked list implementation.', '2024-10-05', 'completed'),
(3, 'create a responsive webpage using html and css.', '2024-10-10', 'pending');
```
```

Output:

| | | | | | | |
|----------------|---------|-------|---------|---------|---------|-------------|
| Administration | Schemas | Users | Courses | Modules | Modules | Leaderboard |
|----------------|---------|-------|---------|---------|---------|-------------|

SCHEMAS

Filter objects

- books_table
- fsd61wde
- GuviZenClassDB
 - Tables
 - Courses
 - LeaveApplications
 - Modules
 - Queries
 - Tasks
 - Users

```

1 • SELECT * FROM Tasks;
2

```

160% 1:1

Result Grid Filter Rows: Search Edit: Export/Import:

| task_id | module_id | task_description | due_date | status |
|---------|-----------|---|------------|-----------|
| 1 | 1 | Complete the coding assignment on variables. | 2024-10-01 | Pending |
| 2 | 2 | Submit the linked list implementation. | 2024-10-05 | Completed |
| 3 | 3 | Create a responsive webpage using HTML and CSS. | 2024-10-10 | Pending |
| NULL | NULL | NULL | NULL | NULL |

List all tasks that are currently pending along with their due dates:

| | | | | | |
|----------------|---------|-------|---------|---------|---------|
| Administration | Schemas | Users | Courses | Modules | Modules |
|----------------|---------|-------|---------|---------|---------|

SCHEMAS

Filter objects

- books_table
- fsd61wde
- GuviZenClassDB
 - Tables
 - Courses
 - LeaveApplications
 - Modules
 - Queries
 - Tasks
 - Users

```

1 • select task_description, due_date
2   from tasks where status = 'pending';
3

```

160% 1:2

Result Grid Filter Rows: Search Export:

| task_description | due_date |
|---|------------|
| Complete the coding assignment on variables. | 2024-10-01 |
| Create a responsive webpage using HTML and CSS. | 2024-10-10 |

Queries:

```

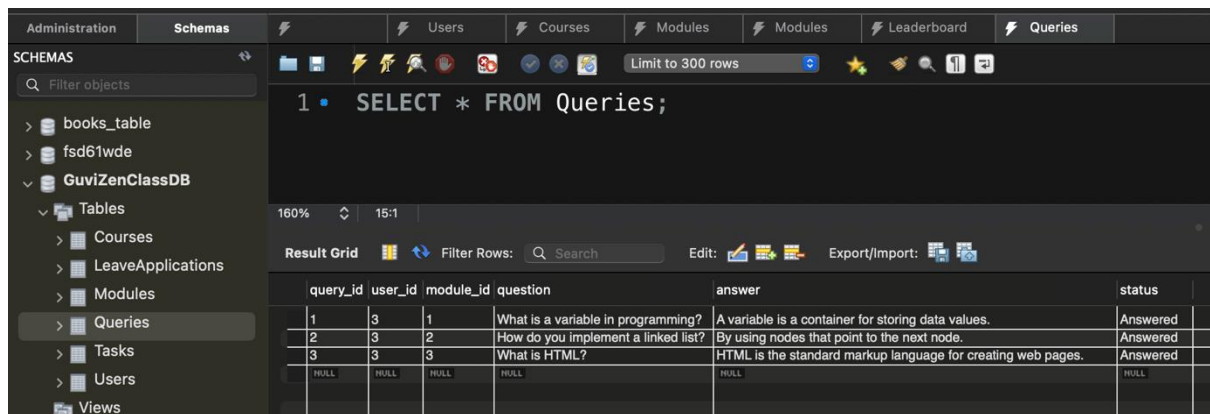
```sql
create table queries (
 query_id int not null auto_increment,
 user_id int not null,
 module_id int not null,
 question text,
 answer text,
 status varchar(20),
 primary key (query_id),
 foreign key (user_id) references users(user_id),

```

```
foreign key (module_id) references modules(module_id)
);
```

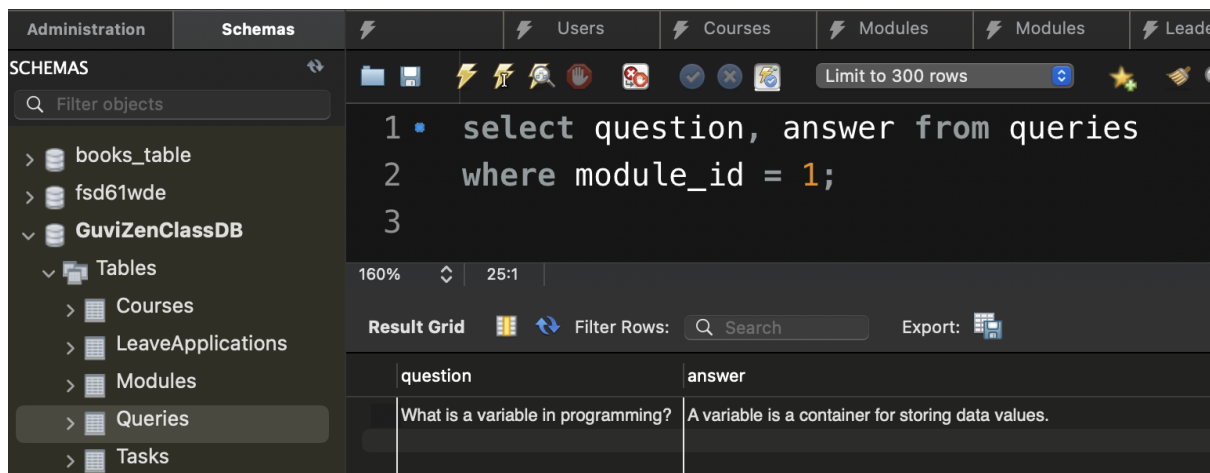
```
insert into queries (user_id, module_id, question, answer, status) values
(3, 1, 'what is a variable in programming?', 'a variable is a container for storing
data values.', 'answered'),
(3, 2, 'how do you implement a linked list?', 'by using nodes that point to the
next node.', 'answered'),
(3, 3, 'what is html?', 'html is the standard markup language for creating web
pages.', 'answered');
```
```

Output:



| query_id | user_id | module_id | question | answer | status |
|----------|---------|-----------|-------------------------------------|--|----------|
| 1 | 3 | 1 | What is a variable in programming? | A variable is a container for storing data values. | Answered |
| 2 | 3 | 2 | How do you implement a linked list? | By using nodes that point to the next node. | Answered |
| 3 | 3 | 3 | What is HTML? | HTML is the standard markup language for creating web pages. | Answered |
| | | | | | |

Retrieve all the questions and their corresponding answers for a specific module:



| question | answer |
|------------------------------------|--|
| What is a variable in programming? | A variable is a container for storing data values. |

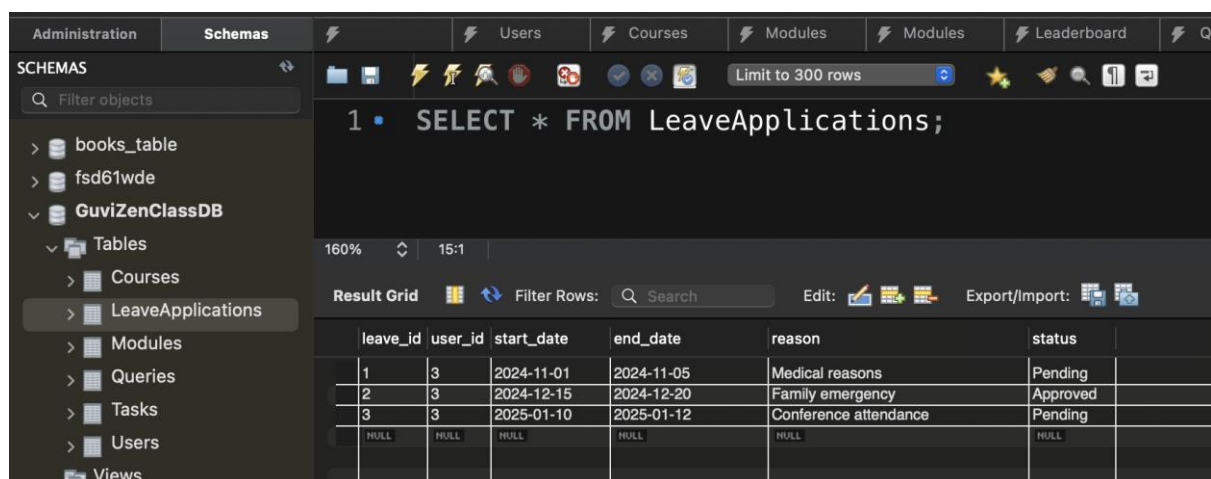
LeaveApplication:

```
```sql
create table leaveapplications (
 leave_id int not null auto_increment,
 user_id int not null,
 start_date date not null,
 end_date date not null,
 reason text,
 status varchar(20) default 'pending',
 primary key (leave_id),
 foreign key (user_id) references users(user_id)
);

insert into leaveapplications (user_id, start_date, end_date, reason, status)
values
(3, '2024-11-01', '2024-11-05', 'medical reasons', 'pending'),
(3, '2024-12-15', '2024-12-20', 'family emergency', 'approved'),
(3, '2025-01-10', '2025-01-12', 'conference attendance', 'pending');
``
```

### Output:

---



The screenshot shows a database management interface with a sidebar on the left displaying a schema tree. The main area shows a SQL query: `SELECT * FROM LeaveApplications;` and a result grid below it. The result grid contains three rows of data, matching the insert statements in the code block above. The columns are: leave\_id, user\_id, start\_date, end\_date, reason, and status. The first row shows leave\_id 1, user\_id 3, start\_date 2024-11-01, end\_date 2024-11-05, reason 'Medical reasons', and status 'Pending'. The second row shows leave\_id 2, user\_id 3, start\_date 2024-12-15, end\_date 2024-12-20, reason 'Family emergency', and status 'Approved'. The third row shows leave\_id 3, user\_id 3, start\_date 2025-01-10, end\_date 2025-01-12, reason 'Conference attendance', and status 'Pending'.

leave_id	user_id	start_date	end_date	reason	status
1	3	2024-11-01	2024-11-05	Medical reasons	Pending
2	3	2024-12-15	2024-12-20	Family emergency	Approved
3	3	2025-01-10	2025-01-12	Conference attendance	Pending



*List all leave applications that have been approved:*

---

The screenshot shows a database management interface with a sidebar on the left labeled 'SCHEMAS' containing a tree view of database objects. The main area displays a SQL query in a text editor, which has been executed. Below the query, a 'Result Grid' shows the output of the query. The query is: `select leave_id, start_date, end_date, reason from leaveapplications where status = 'approved';`. The result grid shows one row of data with columns: leave\_id, start\_date, end\_date, and reason. The data row shows leave\_id 2, start\_date 2024-12-15, end\_date 2024-12-20, and reason Family emergency. There are also rows with NULL values.

Administration Schemas Users Courses Modules Modules Leaderboard Qu

SCHEMAS

Filter objects

books\_table

fsd61wde

GuviZenClassDB

Tables

Courses

LeaveApplications

Modules

Queries

Tasks

Users

Views

1 • select leave\_id, start\_date, end\_date, reason

2 from leaveapplications

3 where status = 'approved';

4

160% 1:4

Result Grid Filter Rows: Search Edit: Export/Import:

leave_id	start_date	end_date	reason
2	2024-12-15	2024-12-20	Family emergency
NULL	NULL	NULL	NULL

## // Join Clause

List all approved leave applications along with the full name of the user who applied for the leave?

The screenshot shows a database management tool interface. On the left, the 'SCHEMAS' panel displays a tree view of the database structure, including tables like 'Users'. The main editor displays a SQL query:

```
1 • use GuviZenClassDB;
2
3 • select
4 u.first_name,
5 u.last_name,
6 l.start_date,
7 l.end_date
8 from
9 leaveapplications l
10 join
11 users u
12 on
13 l.user_id = u.user_id
14 where
15 l.status = 'approved';
16
```

Below the query, the 'Result Grid' shows the results of the query:

first_name	last_name	start_date	end_date
Charlie	Brown	2024-12-15	2024-12-20

List all modules for each course, showing the course name and module name ?

The screenshot shows a database management tool interface. On the left, the 'SCHEMAS' panel displays a tree view of the database structure, including tables like 'Users'. The main editor displays a SQL query:

```
1 • use GuviZenClassDB;
2
3 • select
4 c.course_name,
5 m.module_name
6 from
7 courses c
8 join
9 modules m
10 on
11 c.course_id = m.course_id;
12
```

Below the query, the 'Result Grid' shows the results of the query:

course_name	module_name
Introduction to Programming	Introduction to Variables
Data Structures and Algorithms	Linked Lists
Web Development	HTML Basics