Database
Final Project - Room Tenant System

Group Members
Dicha Zelianivan Arkana - 2241720002
Davis Maulana Hermanto -
Muhammad Baihaqi Aulia Asy'ari
Yanuar Thaif Chalil Candra - 2241720004

Contents

1 Introduction

Room Tenant System is an app to help the university staff manage which room is currenly occupied and which are not. The issue with the current implementation in State Polytechnic of Malang is flawed, in that there is no way for anyone to check whether the room is occupied or not other than manually contacting the staff. This issue causes confusions among the students and staff.

This document lays out the database design for the Room Tenant System app. It includes the Entity Relationship Diagram, the database schema, and the queries. The queries are written in SQL and the database is designed for MySQL 8.0.

2 Entity Relationship Diagram

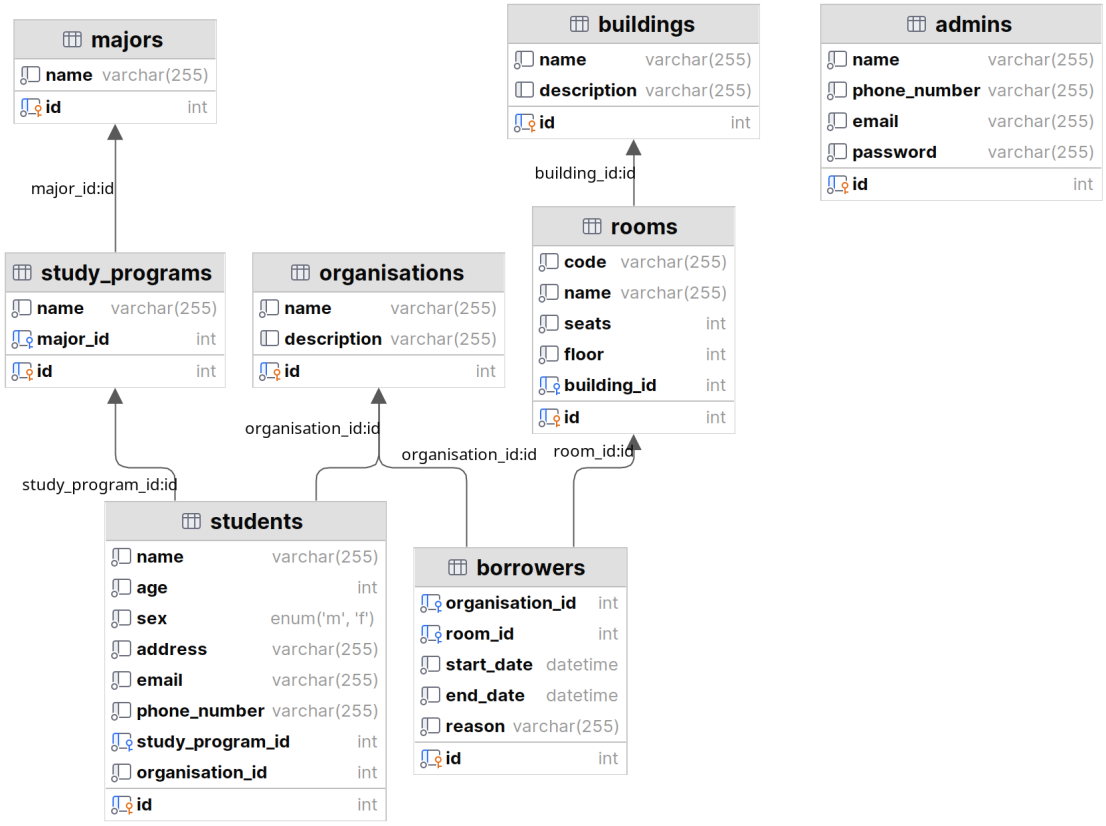


Figure 1: Entity Relationship Diagram of the Room Tenant System

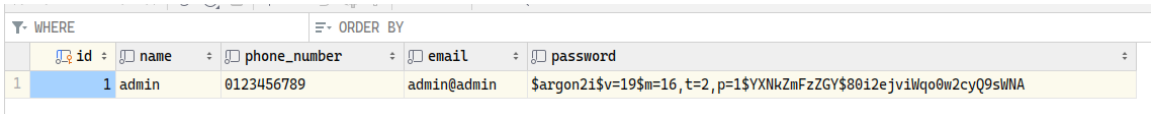
3 Database Schema

The database schema are created using Data Definition Language in SQL. The schema is created to be compatible with MySQL 8.0.

3.1 Admins Table

This table is used to store the data for the admin who will manage the room tenant system in the application. They are the ones who are responsible to review then approve or decline requests.

```
CREATE TABLE `admins`
(
    `id`           INT(11)      NOT NULL AUTO_INCREMENT PRIMARY KEY,
    `name`         VARCHAR(255) NOT NULL,
    `phone_number` VARCHAR(255) NOT NULL,
    `email`        VARCHAR(255) NOT NULL,
    `password`     VARCHAR(255) NOT NULL -- hashed with argon2 algorithm
);
```



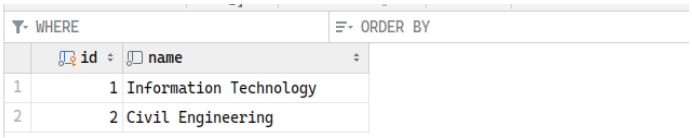
	id	name	phone_number	email	password
1	1	admin	0123456789	admin@admin	$argon2i$v=19$m=16,t=2,p=1$YXNkZmFzZGY$80i2ejviWqo0w2cyQ9sWNA

Figure 2: Admins table with data

3.2 Majors Table

This table is used to store the data for the majors of the university.

```
CREATE TABLE `majors`
(
    `id`   INT(11)      NOT NULL AUTO_INCREMENT PRIMARY KEY,
    `name` VARCHAR(255) NOT NULL
);
```



	id	name
1	1	Information Technology
2	2	Civil Engineering

Figure 3: Majors table with data

3.3 Study Programs Table

This table is used to store the study programs that are available in the university. Each study program will have a major attached to them.

```
CREATE TABLE `study_programs`
(
    `id`       INT(11)      NOT NULL AUTO_INCREMENT PRIMARY KEY,
    `name`     VARCHAR(255) NOT NULL,
    `major_id` INT(11)      NOT NULL
);
```

WHERE ORDER BY

	id	name	major_id
1	1	Informatics Engineering	1
2	2	Construction Engineering	2
3	3	Business Information System	1

Figure 4: Study Programs table with data

3.4 Students Table

This table is used to store the data of the students. Each student stored in the room tenant system will have a study program and an organisation.

```
CREATE TABLE `students`
(
    `id`               INT(11)          NOT NULL AUTO_INCREMENT PRIMARY KEY,
    `nim`              VARCHAR(255)     NOT NULL,
    `name`             VARCHAR(255)     NOT NULL,
    `age`              INT(11)          NOT NULL,
    `sex`              ENUM ('M', 'F') NOT NULL,
    `address`          VARCHAR(255)     NOT NULL,
    `email`            VARCHAR(255)     NOT NULL,
    `phone_number`     VARCHAR(255)     NOT NULL,
    `study_program_id` INT(11)          NOT NULL
);
```

WHERE ORDER BY

	id	nim	name	age	sex	address	email	phone_number	study_program_id
1	1	2241720001	John	20	M	123 Fake Street	john@john	0123456789	1
2	2	2241720002	Jane	18	F	123 Fake Street	jane@jane	0123456789	1
3	3	2241820001	Mary	19	F	123 Fake Street	mary@mary	0123456789	3
4	4	2241820002	Alice	21	F	123 Fake Street	alice@alice	0123456789	3
5	5	2241820003	Bob	20	M	123 Fake Street	bob@bob	0123456789	3
6	6	2241720003	Charlie	19	M	123 Fake Street	charlie@charlie	0123456789	1
7	7	2241820004	David	20	M	123 Fake Street	david@david	0123456789	3
8	8	2241820005	Eve	22	F	123 Fake Street	eve@eve	0123456789	3
9	9	2241720004	Frank	18	M	123 Fake Street	frank@frank	0123456789	1
10	10	2241720005	George	19	M	123 Fake Street	george@george	0123456789	1

Figure 5: Students table with data

3.5 Organisations Table

This table is used to store the data for the student's organisation. Only an organisation can submit a request to borrow a room.

```
CREATE TABLE `organisations`
(
    `id`          INT(11)      NOT NULL AUTO_INCREMENT PRIMARY KEY,
    `name`        VARCHAR(255) NOT NULL,
    `description` VARCHAR(255)
);
```

WHERE ORDER BY

	id	name	description
1	1	Informatics Research Workshop	An organisation that focuses in informatics research and development
2	2	Information Technology Department English Community	An organisation that foster students who are interested in the English language

Figure 6: Organisations table with data

3.6 Students Organisations Table

This table act as a pivot table that connects the many to many relationship between the student and an organisation. Many to many is used because a single student can join multiple organisations and a single organisations can have multiple student members.

```
CREATE TABLE `students_organisations`
(
    `id`              INT(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    `student_id`      INT(11) NOT NULL,
    `organisation_id` INT(11) NOT NULL
);
```

WHERE ORDER BY

	id	student_id	organisation_id
1	1	1	1
2	2	2	2
3	3	3	1
4	4	4	2
5	5	5	1
6	6	6	2
7	7	7	1
8	8	8	1
9	9	8	2
10	10	9	1
11	11	10	1
12	12	10	2

Figure 7: Students Organisations table with data

3.7 Buildings Table

```
CREATE TABLE `buildings`
(
    `id`          INT(11)      NOT NULL AUTO_INCREMENT PRIMARY KEY,
    `name`        VARCHAR(255) NOT NULL,
    `description` VARCHAR(255)
);
```

WHERE ORDER BY

	id	name	description
1	1	Civil Engineering Building	Building that houses both Civil Engineering and Information Technology Department
2	2	Mechanical Engineering Building	Building that houses the Mechanical Engineering Department

Figure 8: Buildings table with data

3.8 Rooms Table

```
CREATE TABLE `rooms`
(
    `id`          INT(11)      NOT NULL AUTO_INCREMENT PRIMARY KEY,
    `code`        VARCHAR(255) NOT NULL,
    `name`        VARCHAR(255) NOT NULL,
    `seats`       INT(11)      NOT NULL,
    `floor`       INT(11)      NOT NULL,
    `building_id` INT(11)      NOT NULL
);
```

WHERE ORDER BY

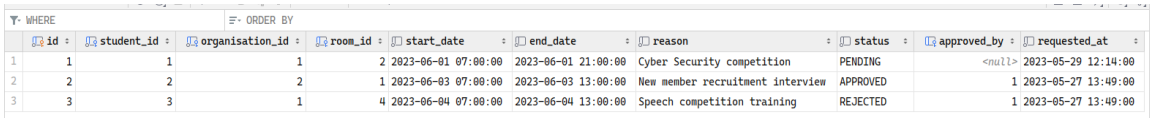
	id	code	name	seats	floor	building_id
1	1	TR-1	Theory Room 1	30	6	1
2	2	PL-1	Project Lab 1	32	7	1
3	3	PL-2	Project Lab 2	24	7	1
4	4	TR-1	Theory Room 1	28	2	2

Figure 9: Rooms table with data

3.9 Borrowed Rooms Table

This table is used to store all of the borrowed rooms. A student can represent an organisation and request a room which will then get reviewed and approved or rejected by the admin.

```
CREATE TABLE `borrowed_rooms`
(
    `id`              INT(11)                                  NOT NULL AUTO_INCREMENT PRIMARY KEY,
    `student_id`      INT(11)                                  NOT NULL,
    `organisation_id` INT(11)                                  NOT NULL,
    `room_id`         INT(11)                                  NOT NULL,
    `start_date`      DATETIME                                 NOT NULL,
    `end_date`        DATETIME                                 NOT NULL,
    `reason`          VARCHAR(255)                             NOT NULL,
    `status`          ENUM('PENDING', 'APPROVED', 'REJECTED') NOT NULL,
    `approved_by`     INT(11)                                  NULL,
    `requested_at`    DATETIME                                 NOT NULL
);
```



WHERE ORDER BY

	id	student_id	organisation_id	room_id	start_date	end_date	reason	status	approved_by	requested_at
1	1	1	1	2	2023-06-01 07:00:00	2023-06-01 21:00:00	Cyber Security competition	PENDING	<null>	2023-05-29 12:14:00
2	2	2	2	1	2023-06-03 07:00:00	2023-06-03 13:00:00	New member recruitment interview	APPROVED	1	2023-05-27 13:49:00
3	3	3	1	4	2023-06-04 07:00:00	2023-06-04 13:00:00	Speech competition training	REJECTED	1	2023-05-27 13:49:00

Figure 10: Borrowed Rooms table with data

3.10 Table Constraints

These are the constraints that is attached to each table. This process uses the `ALTER TABLE` command to add a foreign key to the respective tables. This process is done last to make it more flexible when adding a constraint since both table need to be exists before we can attach the constraint.

```
ALTER TABLE `students`
    ADD FOREIGN KEY (`study_program_id`) REFERENCES `study_programs` (`id`);
ALTER TABLE `study_programs`
    ADD FOREIGN KEY (`major_id`) REFERENCES `majors` (`id`);
ALTER TABLE `students_organisations`
    ADD FOREIGN KEY (`student_id`) REFERENCES `students` (`id`);
ALTER TABLE `students_organisations`
    ADD FOREIGN KEY (`organisation_id`) REFERENCES `organisations` (`id`);
ALTER TABLE `borrowed_rooms`
    ADD FOREIGN KEY (`organisation_id`) REFERENCES `organisations` (`id`);
ALTER TABLE `borrowed_rooms`
    ADD FOREIGN KEY (`room_id`) REFERENCES `rooms` (`id`);
ALTER TABLE `borrowed_rooms`
    ADD FOREIGN KEY (`student_id`) REFERENCES `students` (`id`);
ALTER TABLE `borrowed_rooms`
    ADD FOREIGN KEY (`approved_by`) REFERENCES `admins` (`id`);
ALTER TABLE `rooms`
    ADD FOREIGN KEY (`building_id`) REFERENCES `buildings` (`id`);
```

4 Populating Data

As it stands currently, the database is empty. This section contains the query used to populate the database using dummy data to simulate a real case scenario. This process is also known as *data seeding*.

```
INSERT INTO `admins` (`id`, `name`, `phone_number`, `email`, `password`)
-- real value: `password`, hash salt: `asdfasdf`, algorithm: argon2
VALUES (1, 'admin', '0123456789', 'admin@admin', '$argon2i$v=19$m=16,t=2,p=1$YXNkZmFzZGY$80i2ejviWqo0w2cyQ9sWNA');

INSERT INTO `majors` (`id`, `name`)
VALUES (1, 'Information Technology'),
       (2, 'Civil Engineering');

INSERT INTO `study_programs` (`id`, `name`, `major_id`)
VALUES (1, 'Informatics Engineering', 1),
       (2, 'Construction Engineering', 2),
       (3, 'Business Information System', 1);

INSERT INTO `organisations` (`id`, `name`, `description`)
VALUES (1, 'Informatics Research Workshop', 'An organisation that focuses in informatics research and development'),
       (2, 'Information Technology Department English Community',
       'An organisation that foster students who are interested in the English language');

INSERT INTO `students` (id, nim, name, age, sex, address, email, phone_number, study_program_id)
VALUES (1, '2241720001', 'John', 20, 'M', '123 Fake Street', 'john@john', '0123456789', 1),
       (2, '2241720002', 'Jane', 18, 'F', '123 Fake Street', 'jane@jane', '0123456789', 1),
       (3, '2241820001', 'Mary', 19, 'F', '123 Fake Street', 'mary@mary', '0123456789', 3),
       (4, '2241820002', 'Alice', 21, 'F', '123 Fake Street', 'alice@alice', '0123456789', 3),
       (5, '2241820003', 'Bob', 20, 'M', '123 Fake Street', 'bob@bob', '0123456789', 3),
       (6, '2241720003', 'Charlie', 19, 'M', '123 Fake Street', 'charlie@charlie', '0123456789', 1),
       (7, '2241820004', 'David', 20, 'M', '123 Fake Street', 'david@david', '0123456789', 3),
       (8, '2241820005', 'Eve', 22, 'F', '123 Fake Street', 'eve@eve', '0123456789', 3),
       (9, '2241720004', 'Frank', 18, 'M', '123 Fake Street', 'frank@frank', '0123456789', 1),
       (10, '2241720005', 'George', 19, 'M', '123 Fake Street', 'george@george', '0123456789', 1);

INSERT INTO `students_organisations` (id, student_id, organisation_id)
VALUES (1, 1, 1),
       (2, 2, 2),
       (3, 3, 1),
       (4, 4, 2),
       (5, 5, 1),
       (6, 6, 2),
       (7, 7, 1),
       (8, 8, 1),
       (9, 8, 2),
       (10, 9, 1),
       (11, 10, 1),
       (12, 10, 2);

INSERT INTO `buildings` (`id`, `name`, `description`)
VALUES (1, 'Civil Engineering Building',
        'Building that houses both Civil Engineering and Information Technology Department'),
       (2, 'Mechanical Engineering Building', 'Building that houses the Mechanical Engineering Department');

INSERT INTO `rooms` (`id`, `code`, `name`, `seats`, `floor`, `building_id`)
VALUES (1, 'TR-1', 'Theory Room 1', 30, 6, 1),
       (2, 'PL-1', 'Project Lab 1', 32, 7, 1),
       (3, 'PL-2', 'Project Lab 2', 24, 7, 1),
       (4, 'TR-1', 'Theory Room 1', 28, 2, 2);

INSERT INTO `borrowed_rooms` (id, student_id, organisation_id, room_id, start_date, end_date, reason, approved_by,
```

```
                              status, requested_at)
VALUES (1, 1, 1, 2, '2023-06-01 07:00:00', '2023-06-01 21:00:00', 'Cyber Security competition', NULL, 'PENDING',
        '2023-05-29 12:14:00'),
       (2, 2, 2, 1, '2023-06-03 07:00:00', '2023-06-03 13:00:00', 'New member recruitment interview', 1, 'APPROVED',
        '2023-05-27 13:49:00'),
       (3, 3, 1, 4, '2023-06-04 07:00:00', '2023-06-04 13:00:00', 'Speech competition training', 1, 'REJECTED',
        '2023-05-27 13:49:00');
```

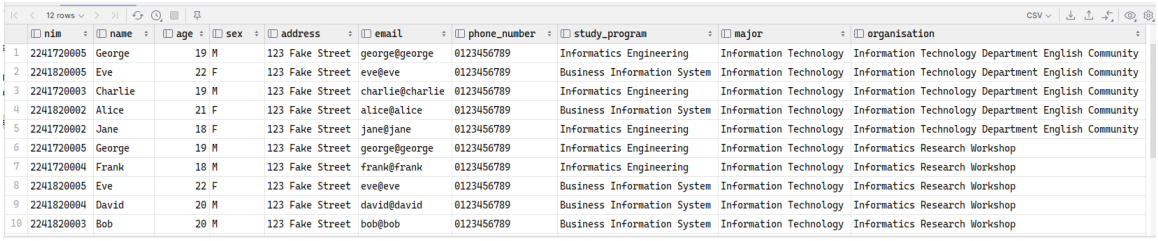
5 Queries

These are some examples of the queries that might be used on the application.

5.1 Students Related Operations

- Get all students along with their major, study program, and organisation

```
SELECT nim,
       std.name,
       age,
       sex,
       address,
       email,
       phone_number,
       sp.name  AS study_program,
       m.name   AS major,
       org.name AS organisation
FROM students_organisations std_org
        JOIN students std ON std_org.student_id = std.id
        JOIN organisations org ON std_org.organisation_id = org.id
        JOIN study_programs sp ON std.study_program_id = sp.id
        JOIN majors m ON sp.major_id = m.id;
```



	nim	name	age	sex	address	email	phone_number	study_program	major	organisation
1	2241720005	George	19	M	123 Fake Street	george@george	0123456789	Informatics Engineering	Information Technology	Information Technology Department English Community
2	2241820005	Eve	22	F	123 Fake Street	eve@eve	0123456789	Business Information System	Information Technology	Information Technology Department English Community
3	2241720003	Charlie	19	M	123 Fake Street	charlie@charlie	0123456789	Informatics Engineering	Information Technology	Information Technology Department English Community
4	2241820002	Alice	21	F	123 Fake Street	alice@alice	0123456789	Business Information System	Information Technology	Information Technology Department English Community
5	2241720002	Jane	18	F	123 Fake Street	jane@jane	0123456789	Informatics Engineering	Information Technology	Information Technology Department English Community
6	2241720005	George	19	M	123 Fake Street	george@george	0123456789	Informatics Engineering	Information Technology	Informatics Research Workshop
7	2241720004	Frank	18	M	123 Fake Street	frank@frank	0123456789	Informatics Engineering	Information Technology	Informatics Research Workshop
8	2241820005	Eve	22	F	123 Fake Street	eve@eve	0123456789	Business Information System	Information Technology	Informatics Research Workshop
9	2241820004	David	20	M	123 Fake Street	david@david	0123456789	Business Information System	Information Technology	Informatics Research Workshop
10	2241820003	Bob	20	M	123 Fake Street	bob@bob	0123456789	Business Information System	Information Technology	Informatics Research Workshop

Figure 11: All students query result

- Get all students from Informatics Engineering

```
SELECT nim,
       std.name,
       age,
       sex,
       address,
       email,
       phone_number,
       sp.name AS study_program
FROM students std
        JOIN study_programs sp ON std.study_program_id = sp.id
WHERE sp.name = 'Informatics Engineering';
```

5 rows

	nim	name	age	sex	address	email	phone_number	study_program
1	2241720001	John	20	M	123 Fake Street	john@john	0123456789	Informatics Engineering
2	2241720002	Jane	18	F	123 Fake Street	jane@jane	0123456789	Informatics Engineering
3	2241720003	Charlie	19	M	123 Fake Street	charlie@charlie	0123456789	Informatics Engineering
4	2241720004	Frank	18	M	123 Fake Street	frank@frank	0123456789	Informatics Engineering
5	2241720005	George	19	M	123 Fake Street	george@george	0123456789	Informatics Engineering

Figure 12: All informatics engineering students query result

- Get all students who requested a room

```
SELECT nim,
       std.name,
       r.name    AS room,
       r.floor   AS floor,
       b.name    AS building,
       br.status AS status,
       br.reason AS reason
FROM borrowed_rooms br
        JOIN students std ON br.student_id = std.id
        JOIN rooms r ON br.room_id = r.id
        JOIN buildings b ON r.building_id = b.id;
```

3 rows

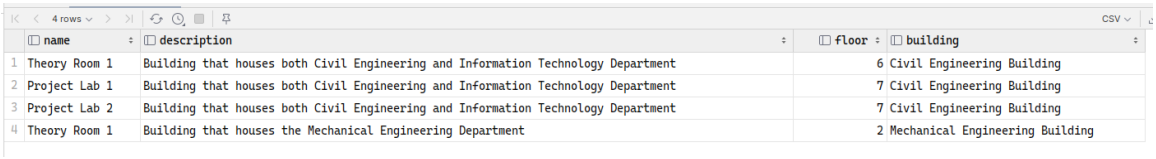
	nim	name	room	floor	building	status	reason
1	2241720001	John	Project Lab 1	7	Civil Engineering Building	PENDING	Cyber Security competition
2	2241720002	Jane	Theory Room 1	6	Civil Engineering Building	APPROVED	New member recruitment interview
3	2241820001	Mary	Theory Room 1	2	Mechanical Engineering Building	REJECTED	Speech competition training

Figure 13: All borrowing students query result

5.2 Rooms Related Operations

- Get all rooms available for borrowing

```
SELECT r.name,
       description,
       floor,
       b.name as building
FROM rooms r
        JOIN buildings b on r.building_id = b.id;
```

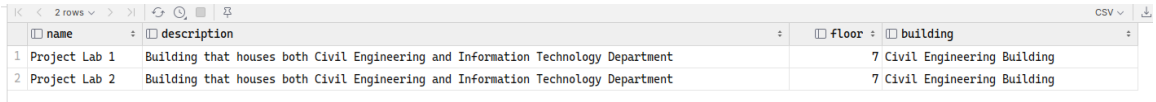


	name	description	floor	building
1	Theory Room 1	Building that houses both Civil Engineering and Information Technology Department	6	Civil Engineering Building
2	Project Lab 1	Building that houses both Civil Engineering and Information Technology Department	7	Civil Engineering Building
3	Project Lab 2	Building that houses both Civil Engineering and Information Technology Department	7	Civil Engineering Building
4	Theory Room 1	Building that houses the Mechanical Engineering Department	2	Mechanical Engineering Building

Figure 14: All rooms query result

- Get all rooms in the 7th floor

```
SELECT r.name,
       description,
       floor,
       b.name as building
FROM rooms r
        JOIN buildings b on r.building_id = b.id
WHERE floor = 7;
```



	name	description	floor	building
1	Project Lab 1	Building that houses both Civil Engineering and Information Technology Department	7	Civil Engineering Building
2	Project Lab 2	Building that houses both Civil Engineering and Information Technology Department	7	Civil Engineering Building

Figure 15: All rooms in 7th floor