

Ahsanullah University of Science and Technology



Department of Computer Science & Engineering

Project Title: Hotel Management System

Course No.	CSE 4126
Course Title	Distributed Database System

Submitted To

Mohammad Imrul Jubair Assistant Professor, Dept. of CSE
Safrun Nesa Saira Lecturer, Dept. of CSE

Submitted By

Name:	Rukaiya Azmi
ID No:	15.01.04.077

Other member's ID:

15.01.04.065
15.01.04.068

Introduction:

Travelling in different places is so easy if we can book the perfect hotel and it will be great when we can book them staying at our home. It can be possible through the hotel website. Managing a hotel is a true 24 hours a day, 7 days a week operation. Managing daily responsibilities, team members, operational performance and the guest experience is a complex balancing act that requires great attention to detail, organization and flexibility. Web-based technology, from personal computing programs to online data storage, has greatly reduced the cost of developing, deploying and maintaining the software and systems that individuals and businesses use daily. That's why we choose it to make the management system easier. Here we tried to make a hotel website using Distributed Database System. The hotel has two branches in Sylhet and Chittagong and a main branch in Dhaka. We tried to make the system more smooth and user friendly.

Project Description:

The name of our project is Hotel Management System. We tried to make a perfect use of Distributed Database System. At first we created an ERD (Entity Relationship Diagram) to design our project. Here we kept two branches of the hotel in two different sites and made Dhaka branch the main branch. All the data and information are distributed in the sites. From a site we can only see the data of that particular site. But from the main branch the server, we can access all the data from all sites. After creating the ERD, we created our necessary tables and inserted data. We kept different data in site and host and created a link between them so host can access the sites. Then some functions and procedures were created. Then we fragmented the tables in two different sites and inserted data according to the fragmentation schema. We also made operator trees and triggers.

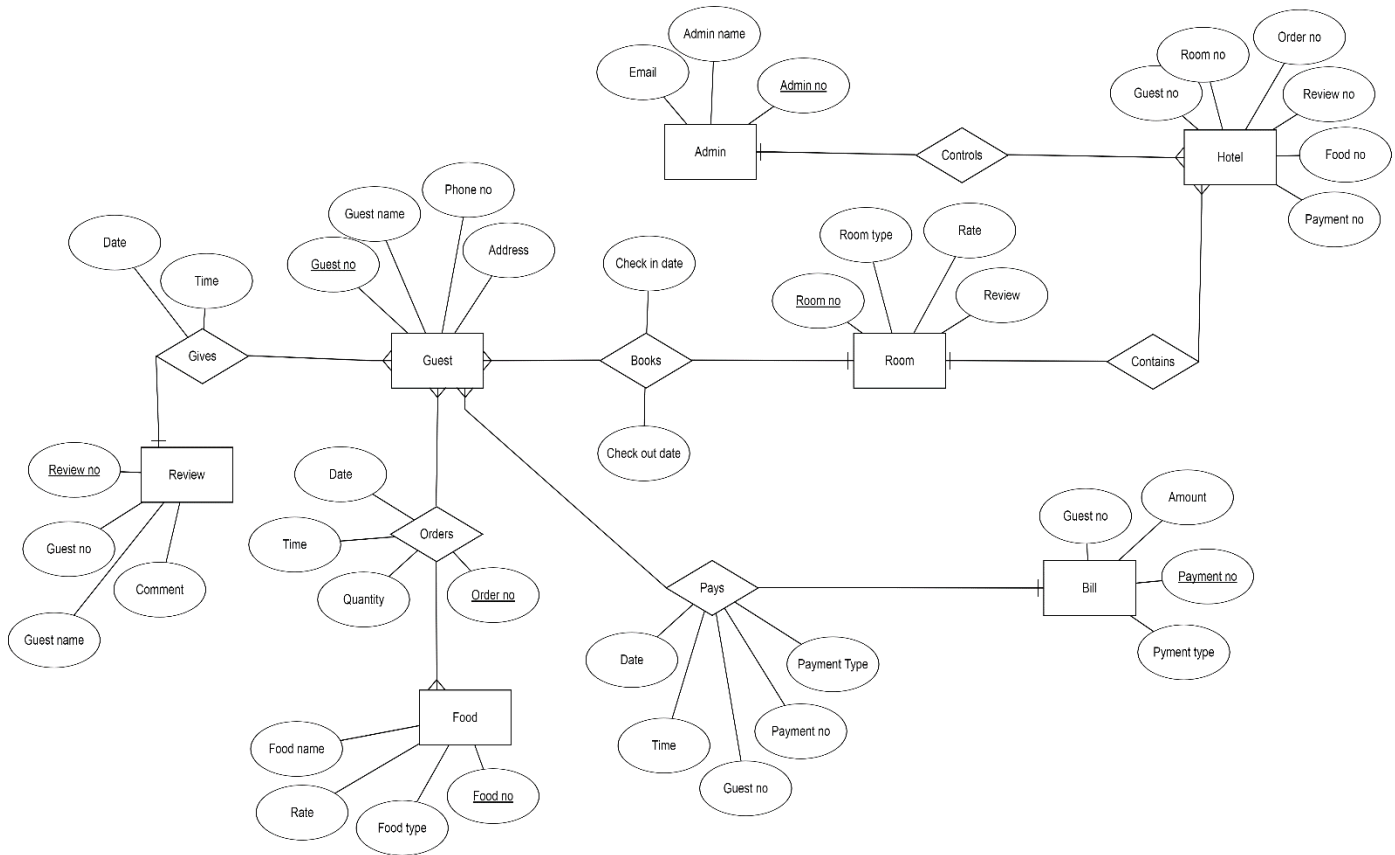
My Contribution:

This was a group project and we are three members in the group. So we divided our works to complete it. In this project, my contribution is-

1. **Creating table and data insertion:** I created seven tables and inserted plenty of data into them.
2. **Database link:** I was there when we created the link. I contributed in the process.
3. **Fragmentation:** We did it together. My laptop was one of the sites and we did this fragmentation using those.
4. **Procedure & Function Creation:** I've created one of the functions & one of the procedures related to this project.
5. **Operator Tree:** I have made an operator tree between the two operator trees is showed in this report. The operator tree from the query, application of criteria and the fragmented tree is done.
6. **Semi-Join Program:** The theory of semi-join program is also proven here. I have shown it with an example.

In this report, I have described all the details of my part of work as well as others' works and contribution but not in descriptively.

Entity Relationship Diagram (ERD):



Entity Relationship Diagram

Database Link:

In database link there are two part host and site. The link is created by myself. First I needed to switch off firewall in both host and site computer. Then I had to check the connection by IP address ping. I have made some slight changes on the listener of the site computer as per instructions. Then I ran the command prompt in administration to see if the listener is working successfully. The host will create a database link and access the tables of the site computer.

Functions and Procedures:

We have created four functions and two procedures for our project. Among them, I made one procedure. The usefulness and name of the procedures are given below:

Functions:

1. **Function1:** Displays the mobile number of the given admin.

```
SQL> @C:\Users\Acer\Desktop\project\Hotel_Management_System\Codes\Function\Function1.sql;  
Function created.  
376  
PL/SQL procedure successfully completed.
```

2. **Function2:** Displays the total employee number of the given room.

```
SQL> @C:\Users\Acer\Desktop\project\Hotel_Management_System\Codes\Function\Function2.sql;  
Function created.  
1  
PL/SQL procedure successfully completed.
```

3. **Function3:** Displays the review of the given room.

```
SQL> @C:\Users\Acer\Desktop\project\Hotel_Management_System\Codes\Function\Function3.sql;  
Function created.  
Excellent  
PL/SQL procedure successfully completed.
```

4. **Function4:** Displays the nationality of the given customer.

```
SQL> @C:\Users\Acer\Desktop\project\Hotel_Management_System\Codes\Function\Function4.sql;  
Function created.  
Bangladeshi  
PL/SQL procedure successfully completed.
```

Procedures:

1. **Procedure1:** Displays the room rate, review, room type of the given room.

```
SQL> @C:\Users\Acer\Desktop\project\Hotel_Management_System\Codes\Procedure\Procedure1.sql;
Procedure created.

8000 Excellent Standard

PL/SQL procedure successfully completed.
```

2. **Procedure2:** Inserts the total amount of the given bill.

```
SQL> @C:\Users\Acer\Desktop\project\Hotel_Management_System\Codes\Procedure\Procedure2.sql;
Procedure created.
```

Database Tables (Global Schema):

1. **Customer** (c_id, name, mobile, email, date_of_birth, nationality)

```
SQL> @"D:\4.1\Lab\DDBS\project\hotel\access for ss.sql";

  C_ID NAME      MOBILE EMAIL      DATE_OF_B NATIONALITY CITY
-----
    1 Laboni      123 laboni@gmail.com 09-OCT-95 Bangladeshi Chittagong
    2 James       234 james@gmail.com 11-JAN-90 American    Chittagong
    3 Prima       345 prima@gmail.com 14-DEC-95 Bangladeshi Sylhet
    4 Kavin       456 kavin@gmail.com 13-APR-90 American    Sylhet
    5 Sharif      567 sharif@gmail.com 21-DEC-91 Bangladeshi Dhaka

Commit complete.

SQL>
```

2. **Admin** (admin_id, name, address, email, mobile_no)

```
SQL> @"D:\4.1\Lab\DDBS\project\hotel\access for ss.sql";
```

ADMIN_ID	NAME	ADDRESS	EMAIL	MOBILE_NO
1	Laboni	27-Azimpur	laboni@gmail.com	298
2	Prima	99-Niketon	prima@gmail.com	876
3	Neaz	20-Bakshibzar	neaz@gmail.com	654

```
Commit complete.
```

```
SQL>
```

3. **RoomDetails** (roomNo, roomType, roomRate, review)

```
SQL> @"D:\4.1\Lab\DDBS\project\hotel\access for ss.sql";
```

ROOMNO	ROOMTYPE	ROOMRATE	REVIEW
1	Standard	8000	Excellent
2	Deluxe	12000	Good
3	Superior-Deluxe	14000	Excellent
4	Junior-Suite	16000	Average

```
Commit complete.
```

```
SQL>
```

4. **Reservation** (rsrv_no, c_id, roomNo, checkin, checkout)

```
SQL> @"D:\4.1\Lab\DDBS\project\hotel\access for ss.sql";
```

RSRV_NO	C_ID	ROOMNO	CHECKIN	CHECKOUT
1	2	4	01-JAN-18	06-JAN-18
2	4	1	03-MAR-18	06-MAR-18
3	2	3	07-APR-18	15-APR-18
4	5	2	10-JUN-18	15-JUN-18

```
Commit complete.
```

5. **Employee** (Emp_id, name, dutyHour, roomNo)

```
SQL> @"D:\4.1\Lab\DDBS\project\hotel\access for ss.sql";
```

EMP_ID	NAME	DUTYHOUR	ROOMNO
1	Rahim	8am-2pm	2
2	Karim	2pm-8pm	4
3	Abul	8pm-2am	1
4	Jabbar	2am-8am	3

```
Commit complete.
```

6. **Food** (foodNo, Type, name, rate)

```
SQL> @"D:\4.1\Lab\DDBS\project\hotel\access for ss.sql";
```

FOODNO	TYPE	NAME	RATE
1	Breakfast	Muffin	200
2	Lunch	Steak	1000
3	Lunch	Chicken	300
4	Dinner	Beef	400

```
Commit complete.
```

7. **Bill** (bill_id, c_id, tot_cost)

```
SQL> @"D:\4.1\Lab\DDBS\project\hotel\access for ss.sql";
```

BILL_ID	C_ID	ROOMNO	AMOUNT	PAYMENTSTATUS
1	2	4	10300	Done
2	4	1	25000	Done
3	2	3	15200	Done
4	5	2	30000	Not Done

```
Commit complete.
```

Fragmentation Schema:

Customer1 = SL city= 'Sylhet' PJ c_id, name, mobile, email, date_of_birth, nationality, city (Customer)

C_ID	NAME	MOBILE	EMAIL	DATE_OF_B	NATIONALITY	CITY
3	Prima	345	prima@gmail.com	14-DEC-95	Bangladeshi	Sylhet
4	Kavin	456	kavin@gmail.com	13-APR-90	American	Sylhet

Commit complete.

Customer2 = SL city!= 'Sylhet' PJ c_id, name, mobile, email, date_of_birth, nationality, city (Customer)

C_ID	NAME	MOBILE	EMAIL	DATE_OF_B	NATIONALITY	CITY
1	Laboni	123	laboni@gmail.com	09-OCT-95	Bangladeshi	Chittagong
2	James	234	james@gmail.com	11-JAN-90	American	Chittagong
5	Sharif	567	sharif@gmail.com	21-DEC-91	Bangladeshi	Dhaka

Commit complete.

Admin1 = SL name = 'Prima' PJ admin_id, name, address, email, mobile_no (Admin)

Admin2 = SL name!= 'Prima' PJ admin_id, name, address, email, mobile_no (Admin)

RoomDetails1 = SL roomRate > 12000 PJ roomNo, roomType, roomRate, review (RoomDetails)

RoomDetails2 = SL roomRate <= 12000 PJ roomNo, roomType, roomRate, review (RoomDetails)

Reservation1 = SL cid = 4 PJ rsrv_no, c_id, roomNo, checkin, checkout (Reservation)

Reservation2 = SL cid!= 4 PJ rsrv_no, c_id, roomNo, checkin, checkout (Reservation)

Employee1 = SL name = 'Rahim' PJ Emp_id, name, dutyHour, roomNo (Employee)

Employee2 = SL name!= 'Rahim' PJ Emp_id, name, dutyHour, roomNo (Employee)

Food1 = SL rate >= 400 PJ foodNo, type, name, rate (Food)

Food2 = SL rate < 400 PJ foodNo, type, name, rate (Food)

Bill1 = SL amount >=25000 PJ bill_id, c_id, roomNo, amount, paymentStatus (Bill)

Bill2 = SL amount <25000 PJ bill_id, c_id, roomNo, amount, paymentStatus (Bill)

RoomDetails1 = PJ roomNo, bill_id, amount (RoomDetails JN roomNo = roomNo Bill1)

RoomDetails2 = PJ roomNo, bill_id, amount (RoomDetails JN roomNo = roomNo Bill2)

Database Profile:

Database profiles for customer and room details table were created.

Card (Customer): 5

```
SQL> @"C:\Users\Acer\Desktop\project\Hotel_Management_System\Codes\Database Profile\Database Profile1.sql";
Table dropped.
:
:
Table created.
Card(Customer) = 5
PL/SQL procedure successfully completed.
```

Card (RoomDetails): 4

```
SQL> @"C:\Users\Acer\Desktop\project\Hotel_Management_System\Codes\Database Profile\Database Profile2.sql";
Table dropped.

Table created.
Card(RoomDetails) = 4
PL/SQL procedure successfully completed.
```

From database profile the profile of results of algebraic operations can be estimated.

Database Trigger:

In our project, two triggers were created. One for update operation and one for insert or delete operation. In update operation, if the new room rate is greater than or equal to fourteen thousand and old room rate is less than fourteen thousand then data will be inserted in site one and data will be deleted from site two and vice versa. Similarly, if the new and old room rate is greater than or equal to fourteen thousand, data will only be updated in site one and vice versa. For insert operation, if new capacity is greater than or equal to fourteen thousand then data will be inserted in site one and for delete operation, if the old capacity is greater than or equal to fifty thousand, data will be deleted from site one.

```
SQL> @C:\Users\Acer\Desktop\project\hotel\trigger\trigger.sql;
Trigger created.
SQL>
```

Semi-Join program:

Semi-join program is more profitable than natural join. In our project, semi-join program is implemented and proved that it's superior.

As example:

For joining RoomDetails and Bill1 in site of Bill1:

Using semi-join would be more profitable here than bringing all data from table RoomDetails to site of Bill2.

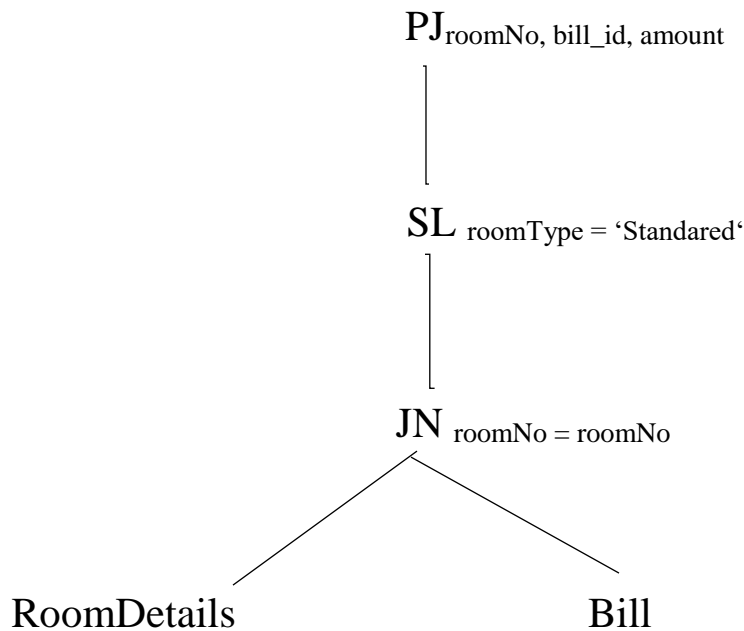
RoomDetails **JN**_{roomNo=roomNo} Bill1 (RoomDetails **SJ**_{roomNo=roomNo} **PJ**_{roomNo} Bill2) **JN**_{roomNo=roomNo} Bill1

Operator Tree:

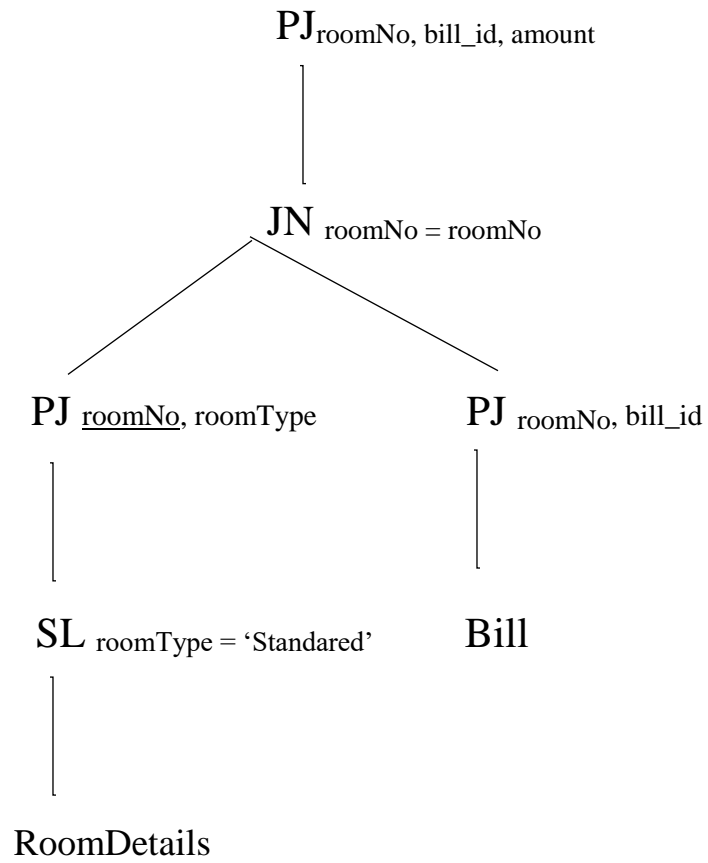
An internal representation of the query is created such as a query tree or query graph. We made two operator trees for our project. I made one of them. Then I derived a fragmentation query from there.

My operator tree query and fragmented query:

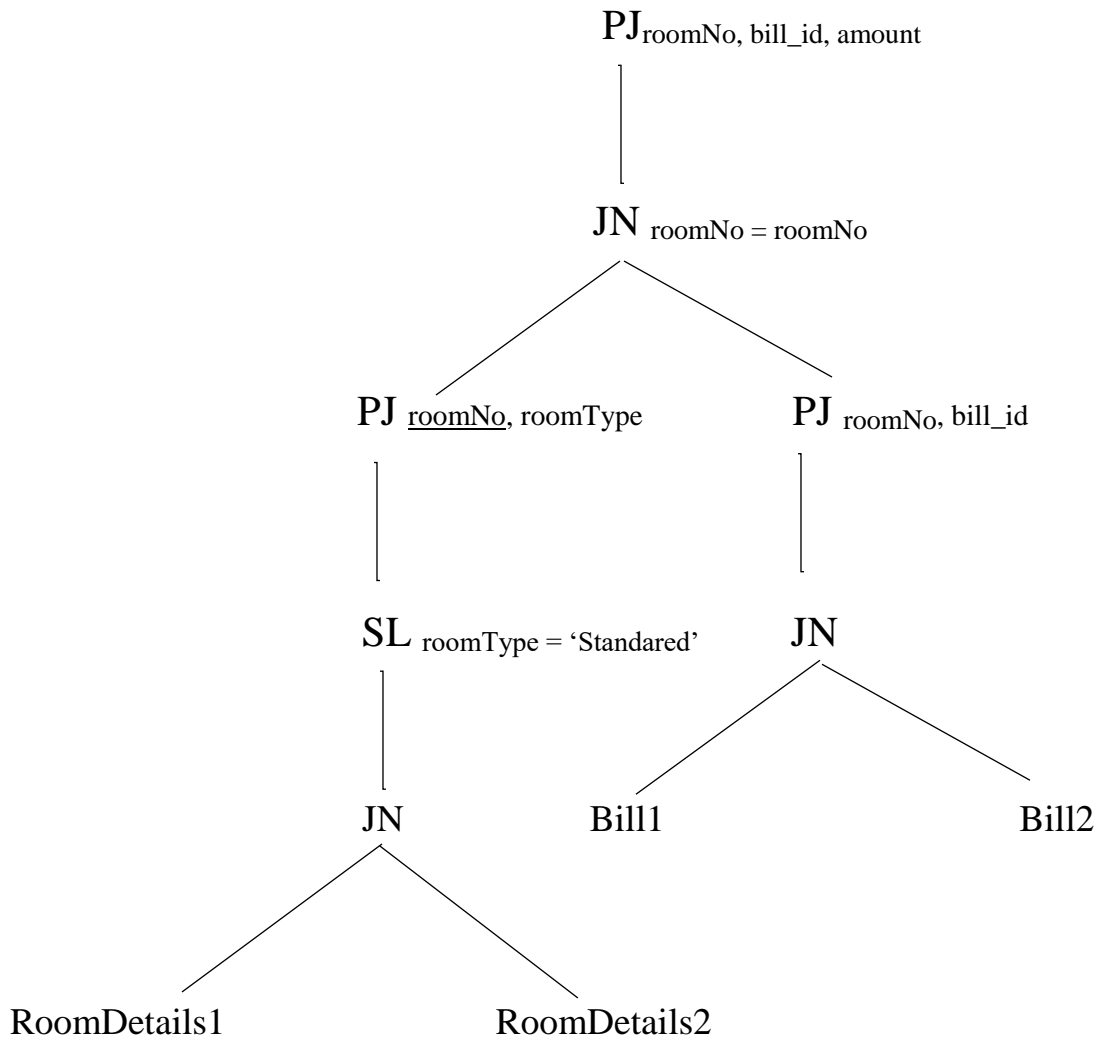
Q2: **PJ**_{roomNo, bill_id, amount} (**SL**_{roomType = 'Standard'} (RoomDetails **JN**_{roomNo = roomNo} Bill))



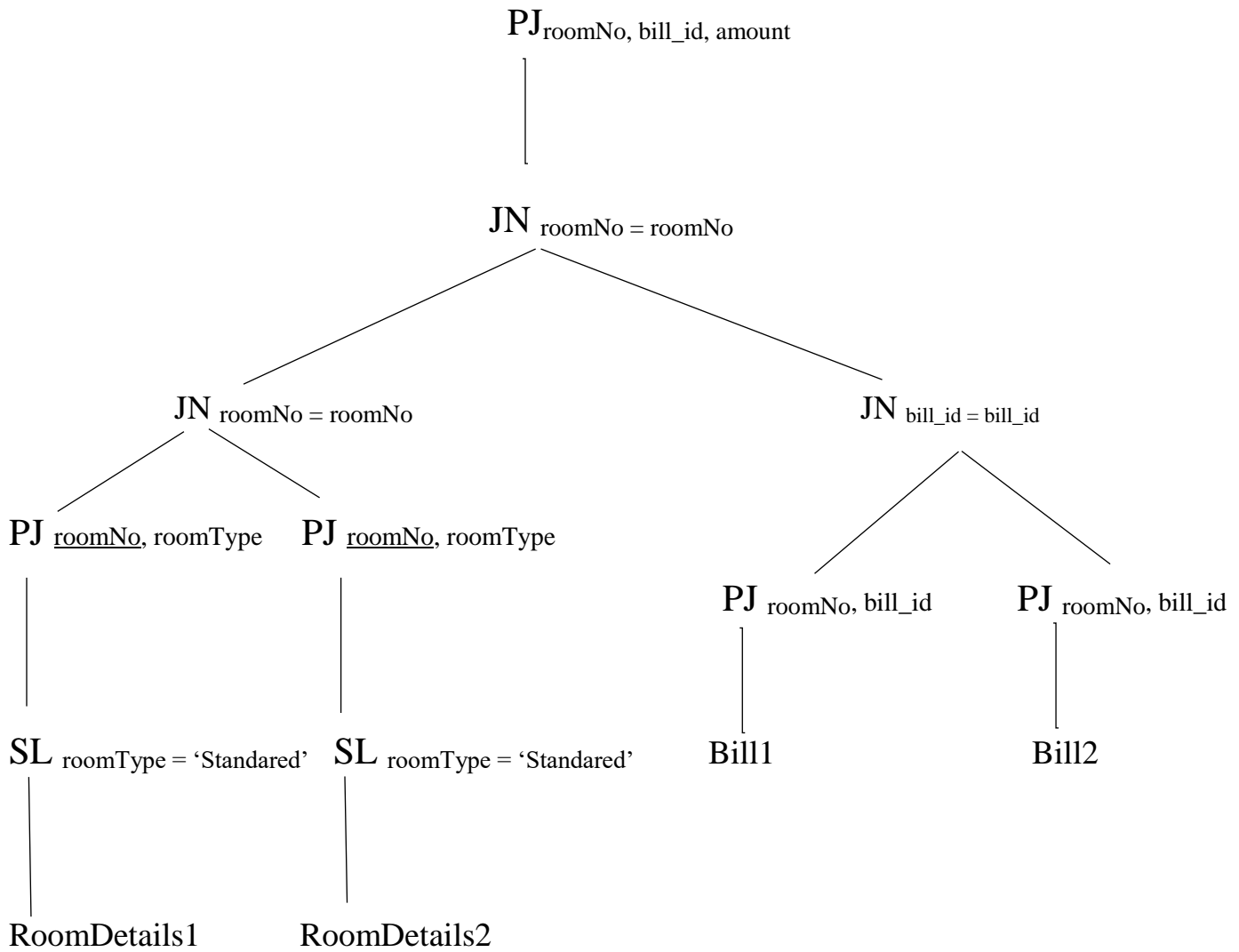
Applying Criterion 1 & 2:



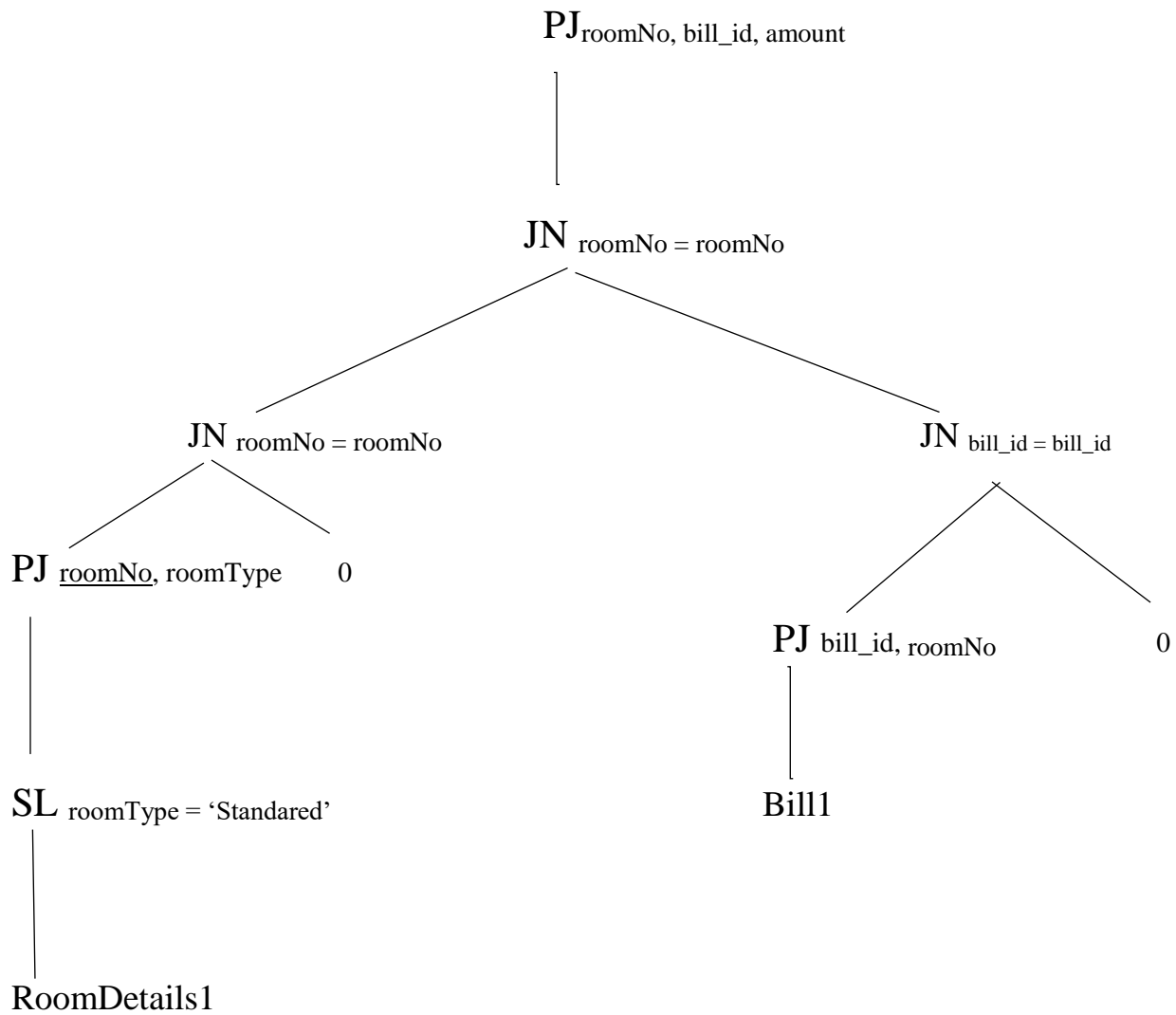
Canonical Expression:



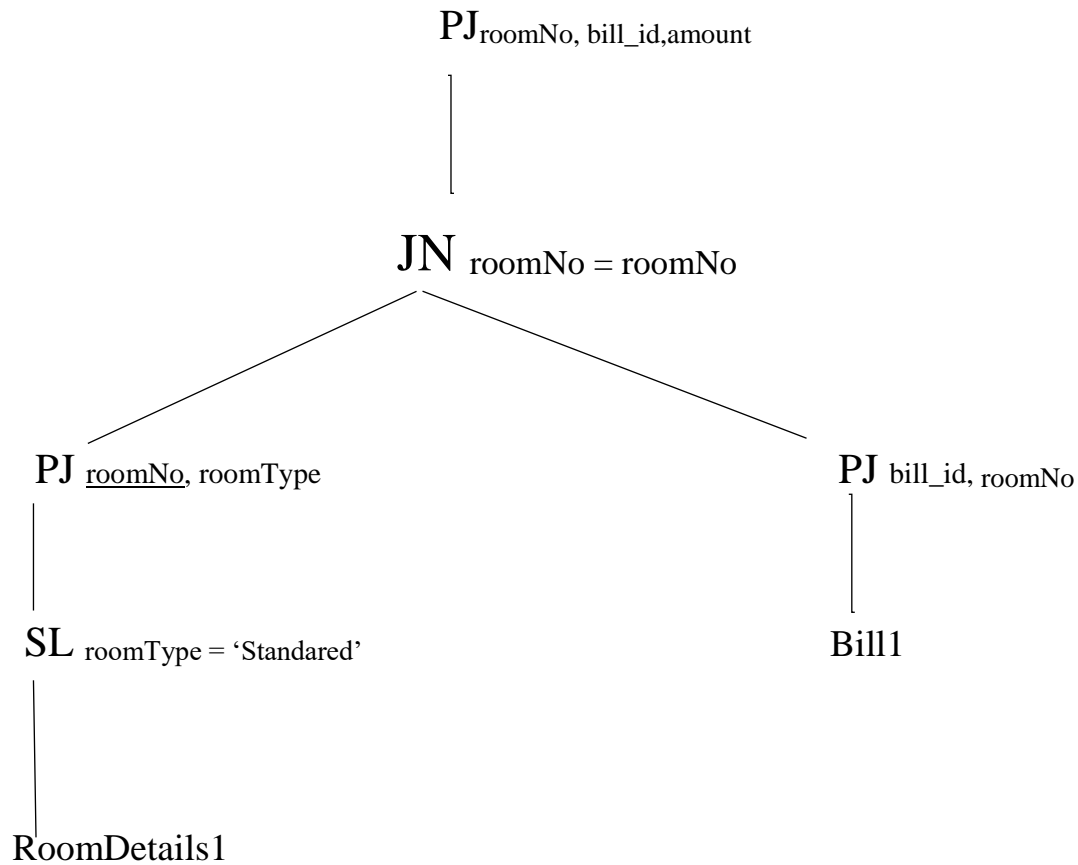
Applying Criteria 1 & 2:



Algebra of Qualified Relation: $SL_F [R: qR] \rightarrow [SL_F R: F \& qR]$



Applying Criteria 3:



Fragmented Query:

QF: PJ roomNo, bill_id, amount ((PJ bill_id, roomNo (Bill1)) JN roomNo = roomNo (PJ roomNo, roomType SL roomType = 'Standarded' (RoomDetails1)))

Conclusion:

We tried our best to make our project as appropriate as we can. There are still many sectors that we can improve and implement more features. In the future, we will try to make it more efficient and make a proper application using this database.