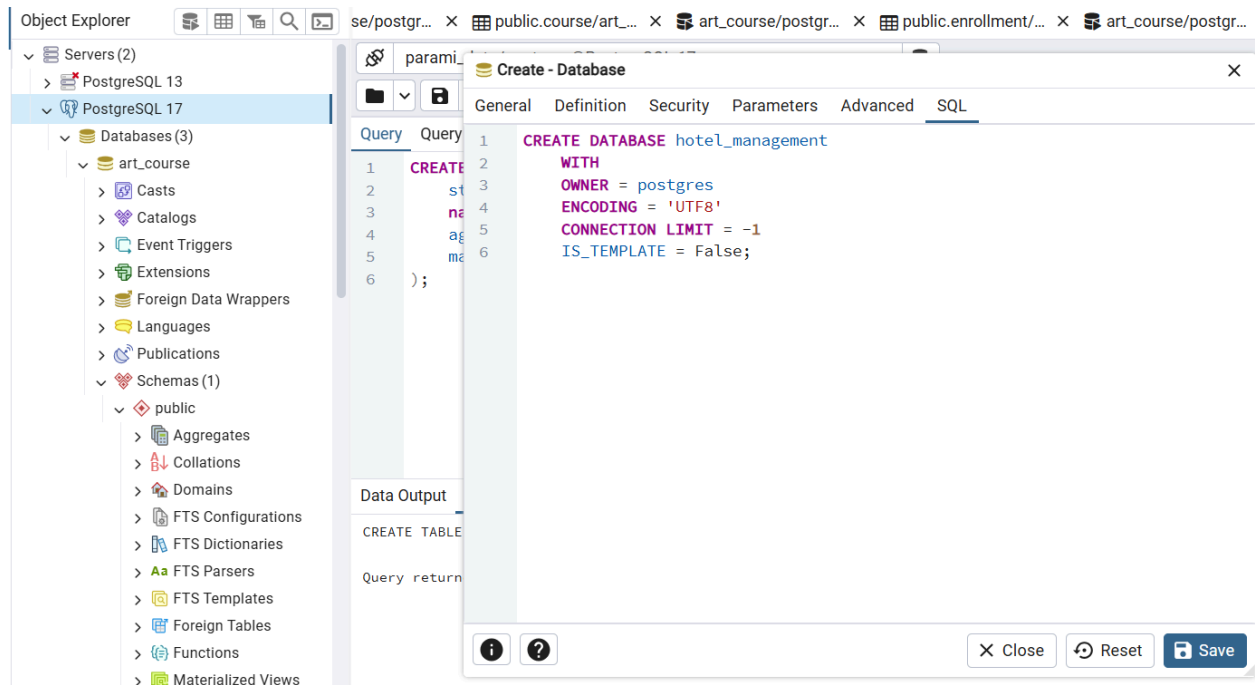


Name: Min Thant Hein
ID: 6805020058

Step 1: Relations Creation

Very first step (Creating a new database):
My database name: 'hotel_management'



#

Creating a new 'Hotel' table:

The screenshot displays a PostgreSQL client interface with a sidebar on the left and a main query editor on the right. The sidebar shows a tree view of the database structure, including 'art_course', 'hotel_management', 'Catalogs', 'Event Triggers', 'Extensions', 'Foreign Data Wrappers', 'Languages', 'Publications', 'Schemas (1)', 'public', 'Aggregates', 'Collations', 'Domains', 'FTS Configurations', 'FTS Dictionaries', 'FTS Parsers', 'FTS Templates', 'Foreign Tables', 'Functions', 'Materialized Views', 'Operators', 'Procedures', 'Sequences', 'Tables (1)', and 'hotel'. The 'hotel' table is selected under 'Tables (1)'. The main query editor shows the following SQL code:

```
1  
2 CREATE TABLE Hotel (  
3     HotelID SERIAL PRIMARY KEY, -- Auto-incrementing primary key  
4     HotelName VARCHAR(100) NOT NULL,  
5     Address TEXT NOT NULL,  
6     City VARCHAR(50) NOT NULL,  
7     Country VARCHAR(50) NOT NULL,  
8     Rating INTEGER CHECK (Rating >= 1 AND Rating <= 5),  
9     Email VARCHAR(100) UNIQUE,  
10    PhoneNumber VARCHAR(20),  
11    CONSTRAINT chk_phone CHECK (PhoneNumber ~ '^[0-9\\-\\+\\(\\)\\s]+$')  
12 );  
13
```

Below the query editor, the 'Data Output' tab is active, showing the message: 'CREATE TABLE' and 'Query returned successfully in 160 msec.'

#

Creating a new 'Room' table,

The screenshot shows a PostgreSQL IDE interface. On the left, the 'Object Explorer' pane displays a tree view of the database structure. The 'hotel_management' database is expanded, showing various objects. The 'Tables (2)' folder is highlighted with a blue box, and the 'room' table is visible below it. The main query editor on the right contains the following SQL code:

```
1 CREATE TABLE Room (  
2     RoomID SERIAL PRIMARY KEY,  
3     HotelID INTEGER NOT NULL REFERENCES Hotel(HotelID) ON DELETE CASCADE,  
4     RoomNo INTEGER NOT NULL CHECK (RoomNo BETWEEN 1 AND 100),  
5     Type VARCHAR(10) NOT NULL CHECK (Type IN ('Single', 'Double', 'Family')),  
6     Price NUMERIC(6,2) NOT NULL CHECK (Price BETWEEN 10 AND 100),  
7     UNIQUE (HotelID, RoomNo) -- Ensures no duplicate room numbers in the same hotel  
8 );  
9
```

Below the query editor, the 'Messages' tab is active, displaying the message: 'Query returned successfully in 160 msec.'

#

Creating a new 'Guest' table:

The screenshot shows a PostgreSQL client interface with a left-hand sidebar displaying a tree view of the database structure. The tree is expanded to show the 'public' schema, and the 'Tables (3)' folder is highlighted with a blue box. Below this, the tables 'guest', 'hotel', and 'room' are listed. The main window displays the SQL query for creating the 'Guest' table, which is executed successfully. The query is as follows:

```
1 CREATE TABLE Guest (  
2     GuestID SERIAL PRIMARY KEY,  
3     FullName VARCHAR(100) NOT NULL,  
4     Email VARCHAR(100) UNIQUE,  
5     PhoneNumber VARCHAR(20),  
6     CONSTRAINT chk_guest_phone CHECK (PhoneNumber ~ '^[0-9\\-\\+\\(\\)\\s]+$')  
7 );  
8  
9
```

The 'Data Output' tab shows the message: 'Query returned successfully in 46 msec.'

#

Creating a new 'Booking' table:

The screenshot shows a PostgreSQL client interface with a sidebar on the left and a main query editor on the right. The sidebar is expanded to show the 'public' schema, where the 'booking' table is highlighted under the 'Tables (4)' category. The main query editor displays the following SQL code:

```
1 CREATE TABLE Booking (  
2     BookingID SERIAL PRIMARY KEY,  
3     HotelID INTEGER NOT NULL REFERENCES Hotel(HotelID) ON DELETE CASCADE,  
4     GuestID INTEGER NOT NULL REFERENCES Guest(GuestID) ON DELETE CASCADE,  
5     RoomNo INTEGER NOT NULL CHECK (RoomNo BETWEEN 1 AND 100),  
6     DateFrom DATE NOT NULL CHECK (DateFrom > CURRENT_DATE),  
7     DateTo DATE NOT NULL CHECK (DateTo > CURRENT_DATE AND DateTo > DateFrom),  
8  
9     -- Prevent double-booking of the same room in the same hotel  
10    CONSTRAINT no_double_booking UNIQUE (HotelID, RoomNo, DateFrom, DateTo),  
11  
12    -- Prevent overlapping bookings for the same guest  
13    CONSTRAINT no_guest_overlap UNIQUE (GuestID, DateFrom, DateTo)  
14 );
```

Below the query editor, the 'Data Output' tab is selected, showing the message: 'Query returned successfully in 64 msec.'

#

##

Step 2: Insert the data using the following SQL (Show the tables in each data)

#Inserting the data into 'Hotel' table:

My Own Note: An error occurred before receiving the following tables. According to the input query lines from the professor, there are only 3 values for 3 columns. However, in the beginning, I created a 'hotel' table with 8 columns. Therefore, an error occurred. Here is the texts indicating error:

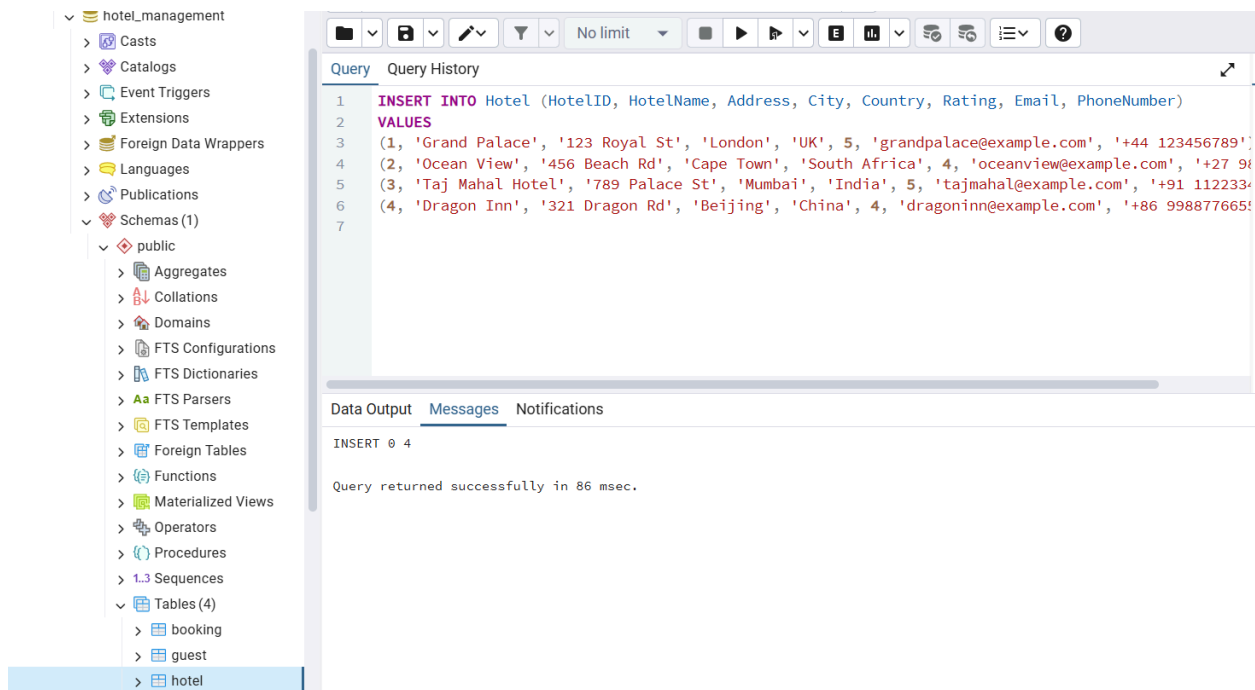
“"ERROR: null value in column "city" of relation "hotel" violates not-null constraint
Failing row contains (1, Grand Palace, London, null, null, null, null, null).”

SQL state: 23502

Detail: Failing row contains (1, Grand Palace, London, null, null, null, null, null).”

Therefore, I had to create and add values for the rest of the columns.

Query, 'Hotel':

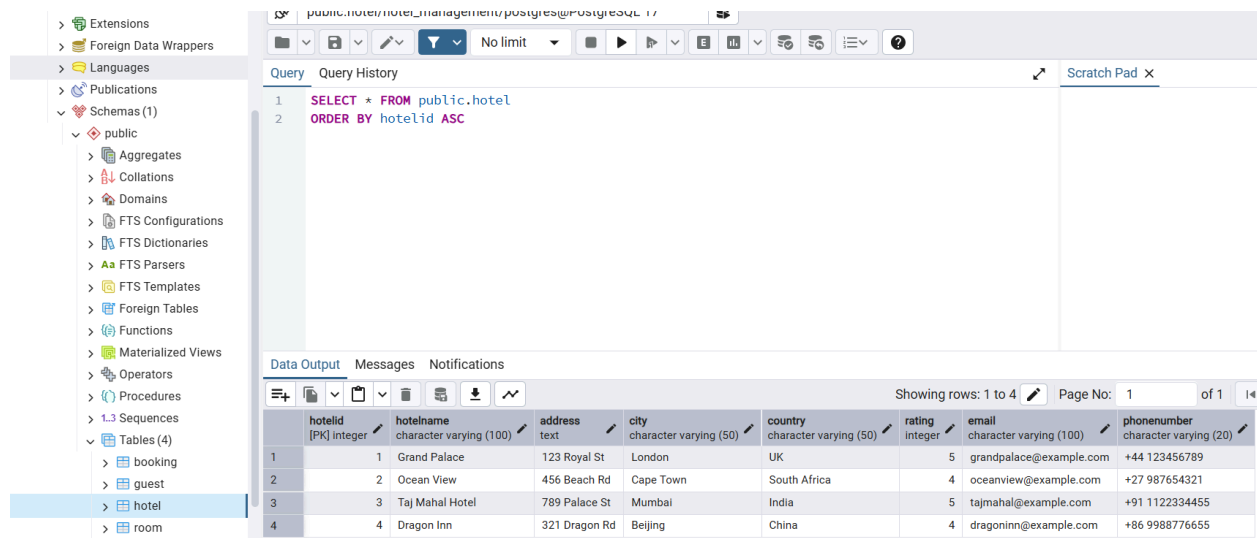


The screenshot displays a database management interface. On the left, a tree view shows the database structure under 'hotel_management', with 'Tables (4)' expanded to show 'booking', 'guest', and 'hotel'. The 'hotel' table is selected. The main panel shows a SQL query editor with the following code:

```
1 INSERT INTO Hotel (HotelID, HotelName, Address, City, Country, Rating, Email, PhoneNumber)
2 VALUES
3 (1, 'Grand Palace', '123 Royal St', 'London', 'UK', 5, 'grandpalace@example.com', '+44 123456789')
4 (2, 'Ocean View', '456 Beach Rd', 'Cape Town', 'South Africa', 4, 'oceanview@example.com', '+27 987654321')
5 (3, 'Taj Mahal Hotel', '789 Palace St', 'Mumbai', 'India', 5, 'tajmahal@example.com', '+91 1122334455')
6 (4, 'Dragon Inn', '321 Dragon Rd', 'Beijing', 'China', 4, 'dragoninn@example.com', '+86 9988776655')
7
```

Below the query editor, the 'Messages' tab is active, showing the execution result: 'INSERT 0 4' and 'Query returned successfully in 86 msec.'

Table view, 'Hotel':



hotelid [FK] integer	hotelname character varying (100)	address text	city character varying (50)	country character varying (50)	rating integer	email character varying (100)	phonenumber character varying (20)
1	Grand Palace	123 Royal St	London	UK	5	grandpalace@example.com	+44 123456789
2	Ocean View	456 Beach Rd	Cape Town	South Africa	4	oceanview@example.com	+27 987654321
3	Taj Mahal Hotel	789 Palace St	Mumbai	India	5	tajmahal@example.com	+91 1122334455
4	Dragon Inn	321 Dragon Rd	Beijing	China	4	dragoninn@example.com	+86 9988776655

#

#Inserting the data into 'Room' Table:

Another error was raised. It is because of mismatch between columns and values. For me, I created 5 columns in total. However, the input values given by the professor, is only for 4 columns. That's why there is a mismatch and having an error. Therefore, I need to include 'RoomNo' and had to match the column order.

Here is the error:

```
" ERROR: invalid input syntax for type integer: "Single"  
LINE 1: INSERT INTO Room VALUES (1, 1, 'Single', 50);  
      ^
```

SQL state: 22P02

Character: 32 "

Query, 'Room':

The screenshot shows the DBeaver database client interface. On the left, the 'Schemas' tree is expanded to 'public', and the 'Tables' folder is selected, showing a list of tables including 'booking', 'guest', 'hotel', and 'room'. The 'hotel' table is highlighted. The main pane displays a SQL query in the 'Query' tab:

```
1 INSERT INTO Room (RoomID, HotelID, RoomNo, Type, Price)
2 VALUES
3 (1, 1, 1, 'Single', 50),
4 (2, 1, 2, 'Double', 80),
5 (3, 2, 1, 'Family', 90),
6 (4, 3, 2, 'Single', 45),
7 (5, 4, 3, 'Double', 70);
8
```

Below the query editor, the 'Messages' tab is active, showing the execution result:

```
INSERT 0 5
Query returned successfully in 99 msec.
```


Table view, 'Room':

The screenshot displays a PostgreSQL client interface with the 'public' schema selected in the left-hand pane. The 'room' table is highlighted under the 'Tables (4)' section. The main pane shows the table's structure and data output.

Table Structure:

	roomid [PK] integer	hotelid integer	roomno integer	type character varying (10)	price numeric (6,2)
--	---------------------	-----------------	----------------	-----------------------------	---------------------

Data Output:

1	1	1	1	Single	50.00
2	2	1	2	Double	80.00
3	3	2	1	Family	90.00
4	4	3	2	Single	45.00
5	5	4	3	Double	70.00

#

#Inserting the data into 'Guest' Table:

Query, 'Guest':

The screenshot displays a PostgreSQL management tool interface. On the left, a sidebar shows the database schema structure, including 'Schemas (1)' and 'public' schema with various objects like 'Aggregates', 'Collations', 'Domains', 'FTS Configurations', 'FTS Dictionaries', 'FTS Parsers', 'FTS Templates', 'Foreign Tables', 'Functions', 'Materialized Views', 'Operators', 'Procedures', 'Sequences', and 'Tables (4)'. The 'Tables (4)' section is expanded, showing 'booking', 'guest', 'hotel', and 'room'.

The main panel shows the 'Query' tab with the following SQL query:

```
1 INSERT INTO Guest VALUES (101, 'Alice Johnson', '123 Main St, London, UK');
2 INSERT INTO Guest VALUES (102, 'Ravi Kumar', 'Mumbai, India');
3 INSERT INTO Guest VALUES (103, 'Chen Wei', 'Beijing, China');
4 INSERT INTO Guest VALUES (104, 'Maria Silva', 'Rio de Janeiro, Brazil');
5 INSERT INTO Guest VALUES (105, 'Andile Nkosi', 'Durban, South Africa');
6
```

The 'Data Output' tab shows the result of the query:

```
INSERT 0 1

Query returned successfully in 83 msec.
```

Table view, 'Guest':

The screenshot shows a database management interface. On the left is a tree view of the database structure. The main area displays a SQL query and its results.

Database Structure (Left Panel):

- hotel_management
 - Casts
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data Wrappers
 - Languages
 - Publications
 - Schemas (1)
 - public
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (4)
 - booking
 - guest
 - hotel
 - room

Query (Main Panel):

```

1  SELECT * FROM public.guest
2  ORDER BY guestid ASC
  
```

Data Output (Table):

	guestid [PK] integer	fullname character varying (100)	email character varying (100)	phonenumber character varying (20)
1	101	Alice Johnson	123 Main St, London, UK	[null]
2	102	Ravi Kumar	Mumbai, India	[null]
3	103	Chen Wei	Beijing, China	[null]
4	104	Maria Silva	Rio de Janeiro, Brazil	[null]
5	105	Andile Nkosi	Durban, South Africa	[null]

#

#Inserting the data into 'Booking' Table:

Another error occurred again. That is not having Hotel_ID in my query which created 'Booking' TABLE. Therefore, I did have to delete the whole 'Room' table and I created another new 'Room' Table which include Hotel_ID/ Hotel_No. Column.

Query, 'Booking':

The screenshot displays a PostgreSQL client interface with the following components:

- Left Panel (Schema Explorer):** A tree view showing the database structure. The 'public' schema is expanded, and the 'booking' table is selected under 'Tables (4)'. Other tables listed are 'guest', 'hotel', and 'room'.
- Top Bar:** Shows the connection name 'hotel_management/postgres@PostgreSQL 17' and various tool icons (connect, save, edit, filter, etc.).
- Query Editor:** Contains the following SQL query:

```
1 INSERT INTO Booking (HotelID, GuestID, RoomNo, DateFrom, DateTo)
2 VALUES
3 (1, 101, 1, '2025-10-20', '2025-10-25'),
4 (3, 102, 2, '2025-11-01', '2025-11-05'),
5 (4, 103, 3, '2025-12-10', '2025-12-20');
6
```
- Bottom Panel:** Divided into three tabs: 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing the execution result:

```
INSERT 0 3

Query returned successfully in 91 msec.
```

Table view, 'Booking':

> Event Triggers

> Extensions

> Foreign Data Wrappers

> Languages

> Publications

> Schemas (1)

> public

> Aggregates

> Collations

> Domains

> FTS Configurations

> FTS Dictionaries

> FTS Parsers

> FTS Templates

> Foreign Tables

> Functions

> Materialized Views

> Operators

> Procedures

> 1.3 Sequences

> Tables (4)

> booking

> guest

> hotel

> room

public.booking/hotel_management/postgres@PostgreSQL 17

No limit

Query Query History

1

2

SELECT * FROM public.booking

ORDER BY bookingid ASC

Data Output Messages Notifications

	bookingid [PK] integer	hotelid integer	guestid integer	roomno integer	datefrom date	dateto date
1	1	1	101	1	2025-10-20	2025-10-25
2	2	3	102	2	2025-11-01	2025-11-05
3	3	4	103	3	2025-12-10	2025-12-20

##

Step 3: Create Views

Query, 'Cheapest hotel view':

The screenshot shows a PostgreSQL IDE interface. On the left is the 'Object Explorer' pane with a tree view of the database schema. The 'public' schema is expanded, and the 'Tables (4)' folder is selected, showing 'booking', 'guest', 'hotel', and 'room'. The 'hotel' table is highlighted. The main editor pane shows a SQL query to create a view named 'CheapestHotels'. The query is as follows:

```
1 CREATE VIEW CheapestHotels AS
2 SELECT h.HotelID, h.HotelName, h.City, MIN(r.Price) AS CheapestPrice
3 FROM Hotel h
4 JOIN Room r ON h.HotelID = r.HotelID
5 GROUP BY h.HotelID, h.HotelName, h.City
6 HAVING MIN(r.Price) = (
7     SELECT MIN(Price) FROM Room
8 );
9
```

Below the query editor, the 'Messages' tab is active, displaying the following message:

```
CREATE VIEW
Query returned successfully in 99 msec.
```

Table view, 'Cheapest hotel view':

rollment/... x art_course/postgr... x parami_data/post... x hotel_manag

public.cheapesthotels/hotel_management/postgres@PostgreS...

Query Query History

```
1 SELECT * FROM public.cheapesthotels
2
```

Data Output Messages Notifications

	hotelid integer	hotelname character varying (100)	city character varying (50)	cheapestprice numeric
1	3	Taj Mahal Hotel	Mumbai	45.00

##

Query, ‘Guests from BRICS countries view’:

The screenshot displays a PostgreSQL client interface with a sidebar on the left and a main query editor on the right. The sidebar shows a tree view of the database schema, including Schemas (1), public, and Views (2). The 'cheapesthotels' view is selected. The main query editor shows a SQL query that creates a view named 'BRICSGuests' based on the 'Guest' table, filtering for guests from BRICS countries (Brazil, Russia, India, China, South Africa) based on their email or phone number. The query is executed, and the results are shown in the 'Data Output' tab, indicating that the query returned successfully in 46 msec.

hotel_management/postgres@PostgreSQL 17

Query Query History

```
1 CREATE VIEW BRICSGuests AS
2 SELECT GuestID, FullName, Email, PhoneNumber
3 FROM Guest
4 WHERE Email ILIKE ANY (
5     ARRAY['%brazil%', '%russia%', '%india%', '%china%', '%southafrica%']
6 )
7 OR PhoneNumber ILIKE ANY (
8     ARRAY['%+55%', '%+7%', '%+91%', '%+86%', '%+27%']
9 );
10
```

Data Output Messages Notifications

CREATE VIEW

Query returned successfully in 46 msec.

Table view, 'Guests from BRICS countries view':

Publications

Schemas (1)

public

Aggregates

Collations

Domains

FTS Configurations

FTS Dictionaries

FTS Parsers

FTS Templates

Foreign Tables

Functions

Materialized Views

Operators

Procedures

1.3 Sequences

Tables (4)

booking

guest

hotel

room

Trigger Functions

Types

Views (2)

bricsguests

cheapesthotels

public.bricsguests/hotel_management/postgres@PostgreSQL...

No limit

Query

Query History

1 SELECT * FROM public.bricsguests

2

Data Output

Messages

Notifications

guestid integer

fullname character varying (100)

email character varying (100)

phonenumber character varying (20)

1 102 Ravi Kumar Mumbai, India [null]

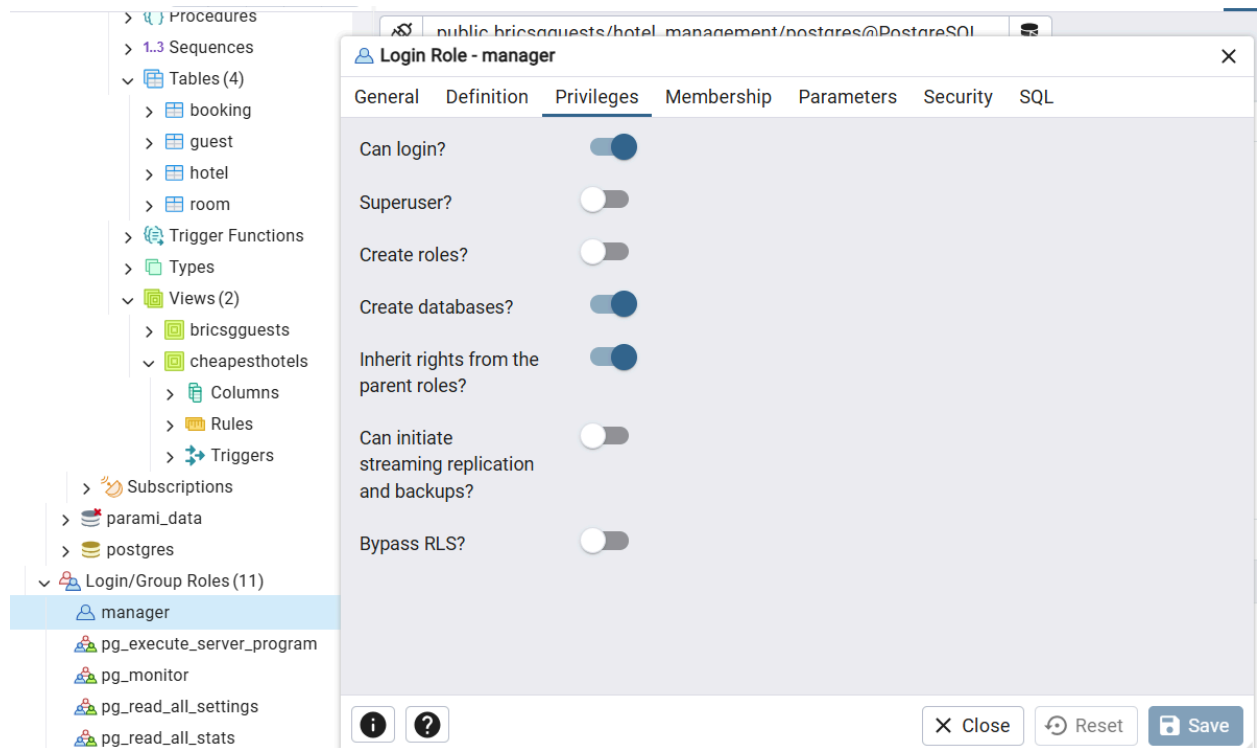
2 103 Chen Wei Beijing, China [null]

3 104 Maria Silva Rio de Janeiro, Brazil [null]

##

Step 4: Create Two User Groups in Pgadmin

#Creating 'manager':



#

#Giving 'manager' view access to both 'bricsgguest' and 'cheapesthotel'

The screenshot displays a PostgreSQL management tool interface. On the left, a tree view shows the database structure, with 'manager' selected under 'Login/Group Roles (11)'. The main panel shows a query editor with the following SQL commands:

```
1 GRANT SELECT ON CheapestHotels TO manager;  
2 GRANT SELECT ON BRICSGGuests TO manager;  
3
```

Below the query editor, the 'Messages' tab is active, showing the output:

```
GRANT  
  
Query returned successfully in 47 msec.
```

#

#Creating 'staff':

The screenshot shows a database management interface. On the left, a tree view lists database objects: Procedures, Sequences, Tables (4) (booking, guest, hotel, room), Trigger Functions, Types, Views (2) (bricsguests, cheapesthotels), Columns, Rules, Triggers, Subscriptions, paraml_data, postgres, and Login/Group Roles (11). The 'Login/Group Roles' folder is expanded, showing roles like manager, pg_execute_server_program, pg_monitor, pg_read_all_settings, pg_read_all_stats, pg_read_server_files, pg_signal_backend, pg_stat_scan_tables, pg_write_server_files, postgres, and 'staff' (selected).

The main window displays the 'Login Role - staff' configuration. The 'Privileges' tab is active, showing a list of permissions with toggle switches:

- Can login? ☒
- Superuser? ☐
- Create roles? ☐
- Create databases? ☒
- Inherit rights from the parent roles? ☒
- Can initiate streaming replication and backups? ☐
- Bypass RLS? ☐

Below the privileges list, there is a 'Data Output' section showing a table with columns 'guestid' and 'integer'. The table has 3 rows of data.

At the bottom right of the configuration window are buttons for 'Close', 'Reset', and 'Save'.

#

#Giving 'staff' view access only to 'Bricsgguest':

> 1.3 Sequences

> Tables (4)

> booking

> guest

> hotel

> room

> Trigger Functions

> Types

> Views (2)

> bricsgguests

> cheapesthotels

> Columns

> Rules

> Triggers

> Subscriptions

> parami_data

> postgres

> Login/Group Roles (11)

manager

pg_execute_server_program

pg_monitor

pg_read_all_settings

pg_read_all_stats

pg_read_server_files

pg_signal_backend

pg_stat_scan_tables

pg_write_server_files

postgres

staff

hotel_management/postgres@PostgreSQL 17

Query

Query History

1

GRANT SELECT ON BRICSGGuests TO staff;

2

Data Output

Messages

Notifications

GRANT

Query returned successfully in 288 msec.

#View permission screenshots, for both, ‘bricsgguest’ and ‘cheapesthotel’ views:

#Both ‘manager’ and ‘staff’ having access to ‘bricsgguest’ view:

The screenshot displays a database management interface with a tree view on the left and a central pane showing the security settings for the 'bricsgguest' view. The 'Privileges' section is expanded, showing a table of permissions.

Grantee	Privileges	Grantor
manager	r	postgres
postgres	Dadrtwx	postgres
staff	r	postgres

The 'Security labels' section is also visible but empty. The bottom of the window shows buttons for 'Close', 'Reset', and 'Save'.

Only 'manager' having access to 'cheapesthotel' view:

FTS Dictionaries

FTS Parsers

FTS Templates

Foreign Tables

Functions

Materialized Views

Operators

Procedures

Sequences

Tables (4)

Trigger Functions

Types

Views (2)

bricsggquests

cheapesthotels

Columns

Rules

Triggers

Query

Query History

1 GRANT SELECT ON B

2

Data Output

Messages

No

GRANT

Query returned successfull

cheapesthotels

General

Definition

Code

Security

SQL

Privileges

	Grantee	Privileges	Grantor
	manager	r	postgres
	postgres	Dadrtwx	postgres

Security labels

Provider	Security label
----------	----------------

Close

Reset

Save

##

Step 5: Test in PSql

#Opening Psql and connecting as 'manager'

```
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:

psql (13.22)
WARNING: Console code page (437) differs from Windows code page (1252)
        8-bit characters might not work correctly. See psql reference
        page "Notes for Windows users" for details.
Type "help" for help.

postgres=# \c hotel_management manager
Password for user manager:

You are now connected to database "hotel_management" as user "manager".
hotel_management=> |
```


#Seeing CheapestHotels View as 'manager' and seeing BRICSGGuests View as 'manager'

```
Username [postgres]:
Password for user postgres:

psql (13.22)
WARNING: Console code page (437) differs from Windows code page (1252)
        8-bit characters might not work correctly. See psql reference
        page "Notes for Windows users" for details.
Type "help" for help.

postgres=# \c hotel_management manager
Password for user manager:

You are now connected to database "hotel_management" as user "manager".
hotel_management=> SELECT * FROM CheapestHotels;
 hotelid | hotelname      | city   | cheapestprice
-----+-----+-----+-----
       3 | Taj Mahal Hotel | Mumbai |          45.00
(1 row)

hotel_management=> SELECT * FROM BRICSGGuests;
 guestid | fullname      | email                | phonenumber
-----+-----+-----+-----
      102 | Ravi Kumar    | Mumbai, India        |
      103 | Chen Wei      | Beijing, China        |
      104 | Maria Silva   | Rio de Janeiro, Brazil |
(3 rows)

hotel_management=> |
```

#Switching to 'staff'

```
WARNING: Console code page (437) differs from Windows code page (1252)
        8-bit characters might not work correctly. See psql reference
        page "Notes for Windows users" for details.
Type "help" for help.
```

```
postgres=# \c hotel_management manager
Password for user manager:
```

```
You are now connected to database "hotel_management" as user "manager".
```

```
hotel_management=> SELECT * FROM CheapestHotels;
```

hotelid	hotelname	city	cheapestprice
3	Taj Mahal Hotel	Mumbai	45.00

```
(1 row)
```

```
hotel_management=> SELECT * FROM BRICSGGuests;
```

guestid	fullname	email	phonenummer
102	Ravi Kumar	Mumbai, India	
103	Chen Wei	Beijing, China	
104	Maria Silva	Rio de Janeiro, Brazil	

```
(3 rows)
```

```
hotel_management=> \c hotel_management staff
Password for user staff:
```

```
You are now connected to database "hotel_management" as user "staff".
```

```
hotel_management=> |
```

#Seeing CheapestHotels View as 'staff' and seeing BRICSGuests View as 'staff'

```
hotel_management=> \c hotel_management staff
Password for user staff:

You are now connected to database "hotel_management" as user "staff".
hotel_management=> SELECT * FROM CheapestHotels;
ERROR:  permission denied for view cheapesthotels
hotel_management=> SELECT * FROM BRICSGuests;
 guestid | fullname | email | phonenumber
-----+-----+-----+-----
      102 | Ravi Kumar | Mumbai, India | 
      103 | Chen Wei | Beijing, China | 
      104 | Maria Silva | Rio de Janeiro, Brazil | 
(3 rows)

hotel_management=> |
```

##