DBMS

HOTEL MANAGEMENT SYSTEM

FINAL PROJECT

Implemented by
Ernar Mukhangali
&
Bexultan Kussainov

Instructor
Mukhtar Amirkumar

PROJECT GOAL

1 Improve hotel performance.

2 Make it easier to manage for owner.

3 Digitalise the hotel system.

PARTS

I
Description
&
Introduction

II
ER diagram

III
Functional
Dependencies

IV
Queries

V
Transactions
&
Indexes

VI
Sources

PART I

Description & Introduction

DESCRIPTION

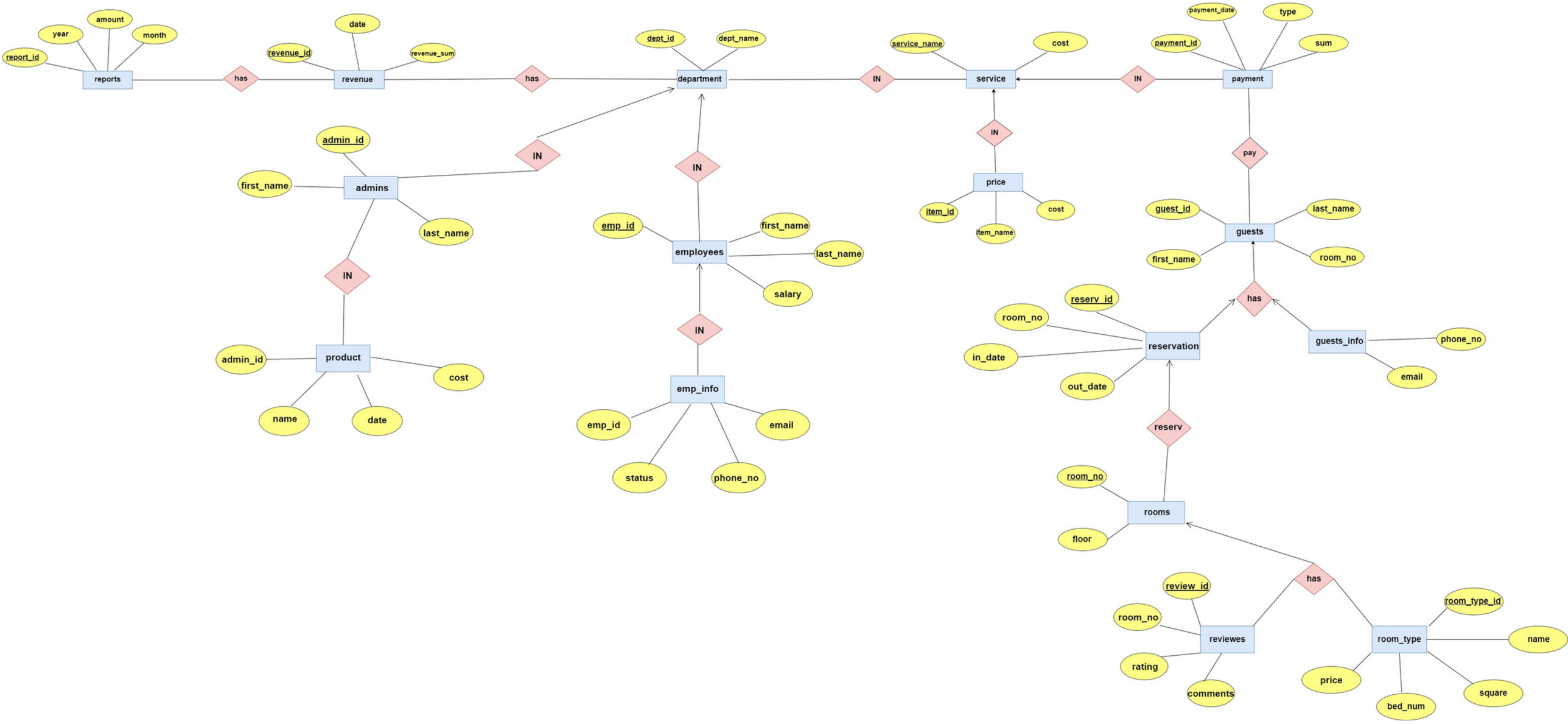
Our database allows the hotel to improve its operations. Makes it easier for the host to manage. The database keeps track of all events within the hotel.

For this we have tables: department, dept_admin, service, employee, employee_info, guest, guest_info, room, room_type, reservation, price, payment, review, revenue, report, product.

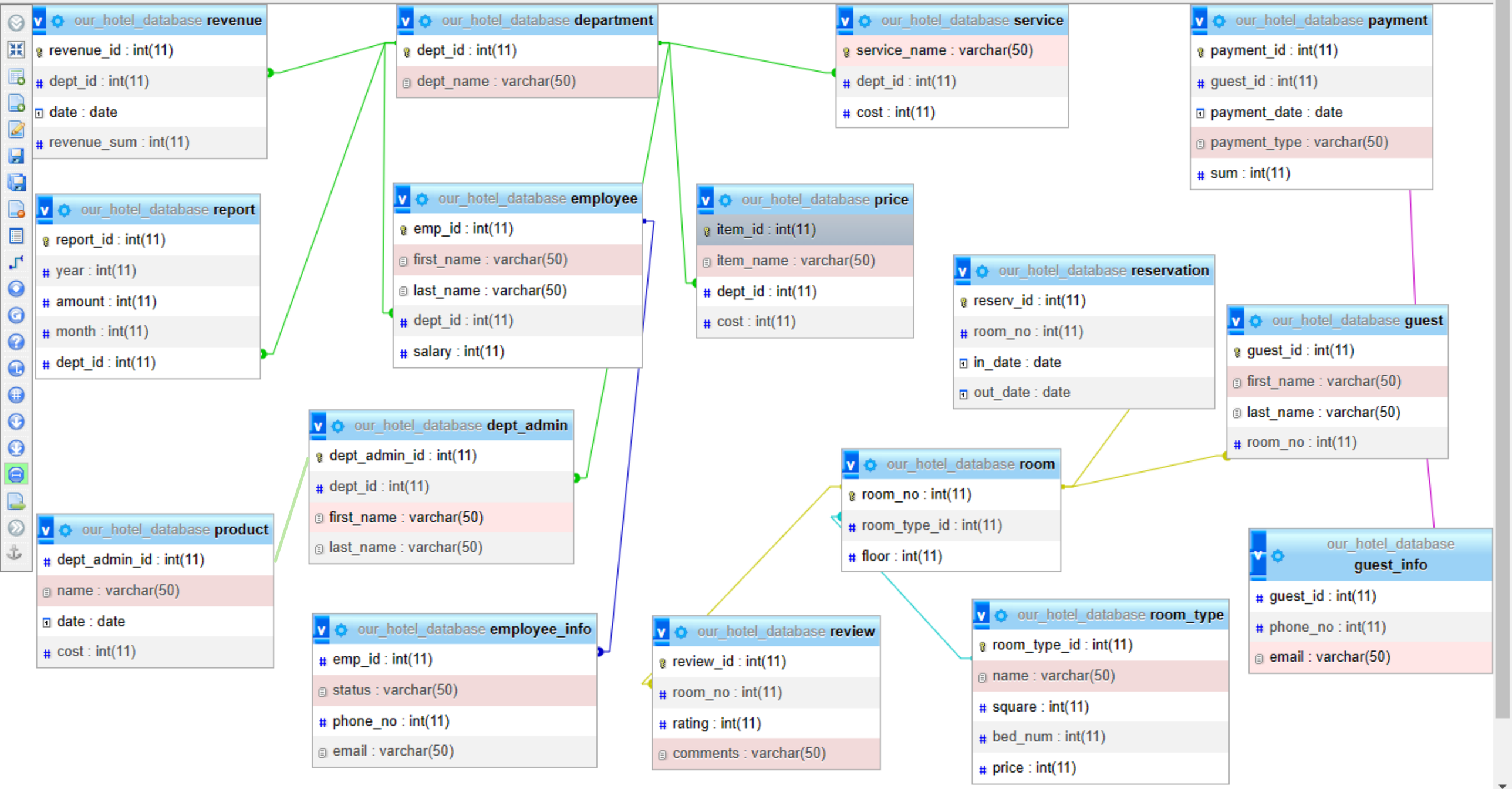
These tables perform the functions of stock control, revenue control, employee performance, guest satisfaction and much more. The end user of the database is the hotelier. The problem of data obsolescence will be solved by triggers. We came up with the idea for the project and the main source of resources was the Internet.

PART II

ER diagram


amount
year
month
report_id
reports
has
date
revenue_id
revenue
revenue_sum
has
dept_id
dept_name
department
IN
service_name
cost
service
IN
payment_date
type
payment_id
sum
payment
admin_id
IN
first_name
admins
last_name
IN
IN
price
item_id
cost
item_name
pay
guest_id
last_name
guests
first_name
room_no
emp_id
first_name
employees
last_name
salary
IN
IN
reserv_id
room_no
has
in_date
reservation
guests_info
phone_no
email
out_date
admin_id
product
cost
name
date
emp_info
emp_id
email
status
phone_no
reserv
room_no
rooms
floor
has
review_id
room_no
reviewes
rating
comments
room_type_id
name
room_type
price
bed_num
square

Untitled *


our_hotel_database revenue
revenue_id : int(11)
dept_id : int(11)
date : date
revenue_sum : int(11)
our_hotel_database department
dept_id : int(11)
dept_name : varchar(50)
our_hotel_database service
service_name : varchar(50)
dept_id : int(11)
cost : int(11)
our_hotel_database payment
payment_id : int(11)
guest_id : int(11)
payment_date : date
payment_type : varchar(50)
sum : int(11)
our_hotel_database report
report_id : int(11)
year : int(11)
amount : int(11)
month : int(11)
dept_id : int(11)
our_hotel_database employee
emp_id : int(11)
first_name : varchar(50)
last_name : varchar(50)
dept_id : int(11)
salary : int(11)
our_hotel_database price
item_id : int(11)
item_name : varchar(50)
dept_id : int(11)
cost : int(11)
our_hotel_database reservation
reserv_id : int(11)
room_no : int(11)
in_date : date
out_date : date
our_hotel_database guest
guest_id : int(11)
first_name : varchar(50)
last_name : varchar(50)
room_no : int(11)
our_hotel_database dept_admin
dept_admin_id : int(11)
dept_id : int(11)
first_name : varchar(50)
last_name : varchar(50)
our_hotel_database room
room_no : int(11)
room_type_id : int(11)
floor : int(11)
our_hotel_database product
dept_admin_id : int(11)
name : varchar(50)
date : date
cost : int(11)
our_hotel_database guest_info
guest_id : int(11)
phone_no : int(11)
email : varchar(50)
our_hotel_database employee_info
emp_id : int(11)
status : varchar(50)
phone_no : int(11)
email : varchar(50)
our_hotel_database review
review_id : int(11)
room_no : int(11)
rating : int(11)
comments : varchar(50)
our_hotel_database room_type
room_type_id : int(11)
name : varchar(50)
square : int(11)
bed_num : int(11)
price : int(11)

$$\pi_{department\ .\ dept\_name,\ AVG\ (cost)}\ \gamma_{AVG\ (cost)}\ \sigma_{department\ .\ dept\_name\ =\ "Housekeeping"}\ (department \bowtie_{department\ .\ dept\_id\ =\ service\ .\ dept\_id} service)$$

$$\pi_{ee\ .\ first\_name,\ ee\ .\ last\_name,\ ef\ .\ phone\_no,\ ef\ .\ email}\ \sigma_{ee\ .\ emp\_id\ =\ ef\ .\ emp\_id}\ (\rho_{ee}\ employee \times \rho_{ef}\ employee\_info)$$

$$\pi_{email}\ guest\_info \cup \pi_{email}\ employee\_info$$

$$\sigma_{payment\_id\ <=\ 5}\ payment \cup \sigma_{p\ .\ payment\_id\ <=\ 5}\ \rho_{p}\ \tau_{payment\ .\ payment\_id\ \downarrow}\ payment$$

$$\gamma_{dept\_id,\ COUNT\ (emp\_id)}\ employee$$

$$\pi_{g\ .\ first\_name,\ g\ .\ last\_name,\ rt\ .\ name}\ (\rho_{g}\ guest \bowtie_{g\ .\ guest\_id\ =\ rt\ .\ room\_type\_id}\ \rho_{rt}\ room\_type)$$

$$\pi_{first\_name,\ last\_name,\ status}\ (\rho_{ee}\ employee \bowtie^{oL}_{ee\ .\ emp\_id\ =\ ef\ .\ emp\_id}\ \rho_{ef}\ employee\_info)$$

PART III

Functional Dependencies

Department

dept_id -> dept_name

Service

service_name -> dept_id , cost

dept_admin

dept_admin_id -> dept_id , first_name,last_name

dept_admin

dept_admin_id -> dept_id , first_name,last_name

guest_info

guest_id -> phone_no

phone_no -> email

guest_id -> email

reservation

reserv_id -> room_no,in_date,out_date

room

room_no -> floor ,room_type_id

employee

emp_id -> dept_id , first_name,last_name , salary

employee_info

emp_id -> status ,email

emp_id -> phone_no

guest

guest_id -> first_name,last_name , room_no

room_type

room_type_id -> name ,square,bed_num,price

review

review_id -> rating

room_no -> comments

review_id -> room_no

product

dept_admin_id -> name,date,cost

payment_id

payment_id -> payment_date

guest_id -> payment_type,sum

payment_id -> guest_id

revenue

revenue_id -> date

dept_id -> revenue_sum

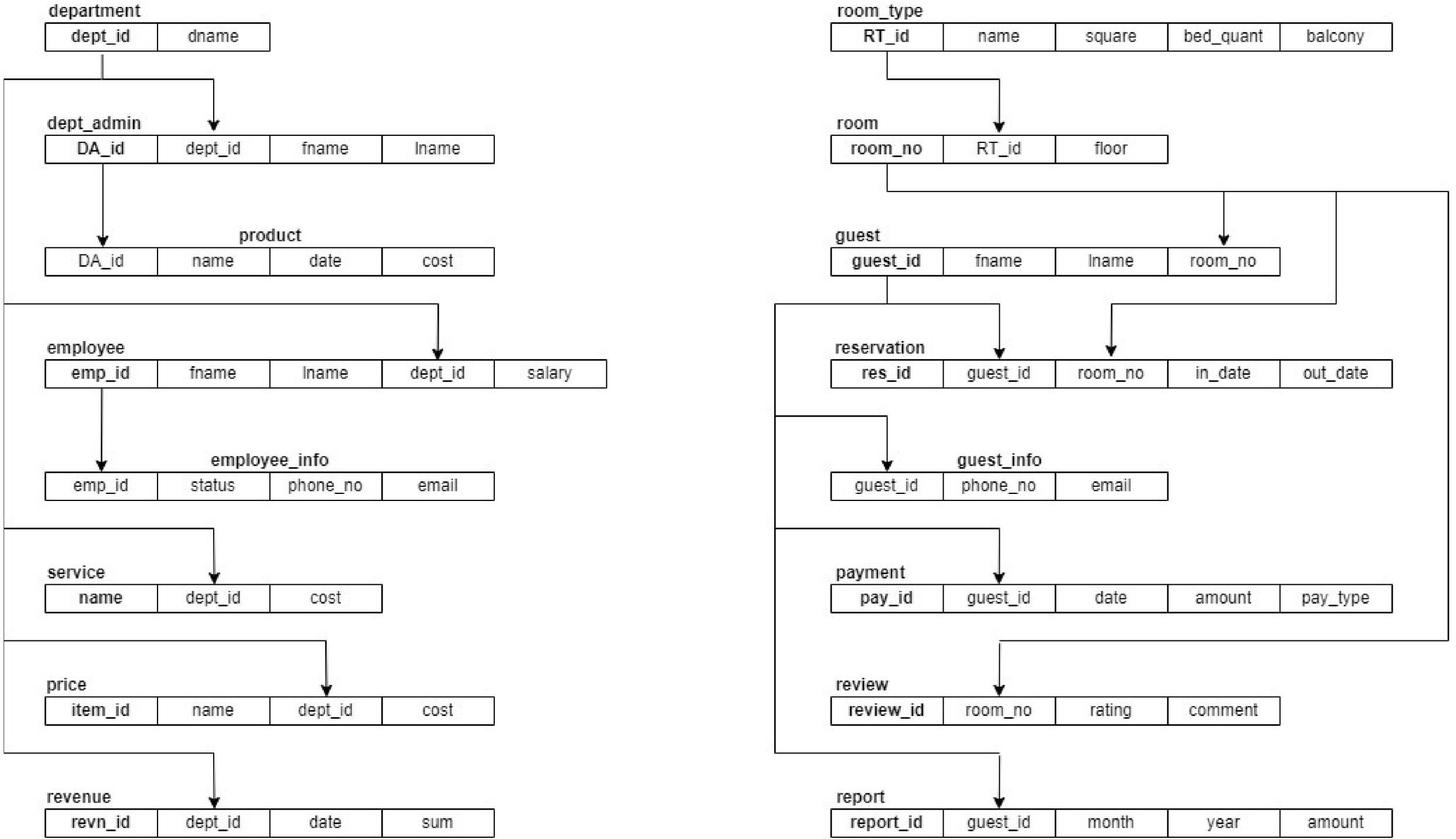
revenue_id -> dept_id

report

report_id -> amount

dept_id -> year,month

report_id -> dept_id


department
dept_id
dname
dept_admin
DA_id
dept_id
fname
lname
product
DA_id
name
date
cost
employee
emp_id
fname
lname
dept_id
salary
employee_info
emp_id
status
phone_no
email
service
name
dept_id
cost
price
item_id
name
dept_id
cost
revenue
revn_id
dept_id
date
sum
room_type
RT_id
name
square
bed_quant
balcony
room
room_no
RT_id
floor
guest
guest_id
fname
lname
room_no
reservation
res_id
guest_id
room_no
in_date
out_date
guest_info
guest_id
phone_no
email
payment
pay_id
guest_id
date
amount
pay_type
review
review_id
room_no
rating
comment
report
report_id
guest_id
month
year
amount

PART IV

Queries

1. Employees' positions by name and last name

```sql
SELECT first_name,last_name,status
FROM employee AS ee
left OUTER JOIN employee_info AS ef on ee.emp_id=ef.emp_id;
```

2. Adds 500 for spa services

```sql
UPDATE price
SET cost = cost+500
where dept_id IN (SELECT dept_id
                  FROM department
                 WHERE dept_name = 'SPA');
```

3. Half of employees

```sql
SELECT *
FROM employee
WHERE  emp_id <=
       (SELECT COUNT(emp_id)/2
        from employee);
```

4. The 5 most recently checked-in guests and their payment

```sql
SELECT * FROM payment
WHERE payment_id <=5
UNION
SELECT * FROM
(SELECT * FROM payment
 ORDER BY payment.payment_id DESC) AS p
 WHERE p.payment_id <=5;
```

5. Guests who have spent less than 500

```sql
SELECT g.first_name,g.last_name,p.sum
from guest g,payment  p
where EXISTS
          (SELECT payment.guest_id,guest.guest_id
           FROM payment,guest
           WHERE guest.guest_id = payment.guest_id AND payment.sum<500);
```

6. Guests by room type

```sql
select g.first_name, g.last_name, rt.name
from guest as g
JOIN room_type as rt
on g.guest_id = rt.room_type_id;
```

7. Guests by their info

```sql
CREATE VIEW employee_information
AS
SELECT  ee.first_name, ee.last_name, ef.phone_no, ef.email
FROM employee as ee, employee_info as ef
WHERE ee.emp_id = ef.emp_id;
```

8. Non-departmentalised services

```sql
Create View newService_name  AS
SELECT service.service_name
FROM service
where service.dept_id Not IN
        (SELECT department.dept_id
         FROM department);
```

9. All emails

```sql
SELECT  email FROM guest_info
UNION
SELECT email FROM employee_info;
```

10. Deletes duplicates

```sql
DELETE e1
FROM employee E1
INNER JOIN employee e2
WHERE e1.emp_id > e2.emp_id
AND e1.first_name = e2.first_name
AND e1.dept_id = e2.dept_id
AND e1.salary=e2.salary;
```

11. Two employees' maximum salary

```sql
SELECT DISTINCT salary
FROM employee e1
WHERE 2 >=
    (SELECT COUNT(DISTINCT salary)
     FROM employee e2
     WHERE e1.salary <= e2.salary)
     ORDER BY e1.salary DESC;
```

12. Departments by number of employees

```sql
SELECT COUNT(emp_id),dept_id
from employee
GROUP BY dept_id;
```

13. Deletes departments with revenues under 1000

```sql
 delete
from dept_admin
     where dept_id in
     (select dept_id
      from revenue
      where sum<1000);
```

14. Departments in terms of monthly reports

```sql
SELECT department.dept_name, report.amount
FROM department
RIGHT JOIN report ON department.dept_id = report.dept_id
ORDER BY report.amount;
```

15. Number of employees in departments

```sql
SELECT department.dept_name , COUNT(employee.emp_id) AS numOfEmp
FROM employee
   LEFT JOIN department ON employee.dept_id = department.dept_id
    GROUP BY dept_name;
```

16. Average cost of housekeeping department's services

```sql
SELECT department.dept_name, AVG(service.cost)
FROM department
JOIN service
       ON department.dept_id = service.dept_id
       WHERE department.dept_name = 'Housekeeping';
```

17. Department, which has the most income

```sql
SELECT revenue_sum,dept_id
FROM revenue r1
WHERE 0 = (
          |    SELECT COUNT( DISTINCT ( r2.revenue_sum ) )
               FROM revenue r2
               WHERE r2.revenue_sum > r1.revenue_sum
              );
```

18. +20% for cost of department services

```sql
DELIMITER $$
CREATE TRIGGER price_update
AFTER UPDATE ON price
FOR EACH ROW
BEGIN
UPDATE price
SET cost = cost + cost * 0.2
WHERE dept_id = (SELECT dept_id
                 FROM department
                 WHERE dept_id = 1);
END $$
DELIMITER ;;
```

19. Deletes department admin

```sql
DELIMITER $$
CREATE TRIGGER After_Delete_admin
AFTER DELETE
ON dept_admin
FOR EACH ROW
BEGIN
    DELETE FROM выручка
    WHERE dept_admin.dept_id = DELETED.dept_id;
END $$
DELIMITER ;;
```

20. +20% for cost of room

```sql
DELIMITER $$
CREATE TRIGGER dsfa
after UPDATE on review
FOR EACH ROW
BEGIN
  if new.rating > old.rating then
      UPDATE room_type set price = price * 1.2;
    end if;
end $$
DELIMITER ;;
```

PART V

Transactions & Indexes

TRANSACTIONS

```
BEGIN TRANSACTION
DELETE dept_name FROM dept_admins
WHERE  dept_name LIKE 'SPA'
DELETE first_name, last_name FROM dept_admins
WHERE admin_id = 3
COMMIT;
```

```
BEGIN TRANSACTION
INSERT INTO employee('101', 'Erlan', 'Jumabayev', '1')
COMMIT;
```

```
BEGIN TRANSACTION
DELETE * FROM employee
WHERE emp_id = 5
DELETE * FROM employee_info
WHERE emp_id = 5
COMMIT;
```

```
BEGIN TRANSACTION
INSERT INTO department('7', 'GYM')
COMMIT;
```

INDEXES

```
CREATE INDEX index_dept_name
ON department(dept_name);
```

```
CREATE INDEX index_emp_info
ON employee_info(phone_no, email);
```

```
CREATE INDEX index_room_rating
ON review(rating);
```

THANK YOU FOR ATTENTION!

SOURCES

Maxat Skakov's DBMS 1 lecture 2.

Maxat Skakov's DBMS 1 lecture 3.

Maxat Skakov's DBMS 1 lecture 4.

Maxat Skakov's DBMS 1 lecture 5.

Maxat Skakov's DBMS 1 lecture 7.

Maxat Skakov's DBMS 1 lecture 8.

Maxat Skakov's DBMS 1 lecture 12.

Maxat Skakov's DBMS 1 lecture 13.