

**Group 20 Project**  
**Bhavajna Kallakuri (202003046)**  
**Divyansh Jain (202003047)**  
**Lab 11 (Date: 15-11-2021)**  
**Bus Management System**

**Views:**

1. View the routeid, source, destination, scheduled date for all the routes

**DDL Code:**

```
CREATE VIEW routes_view AS  
SELECT routeid, source_name, destination_name, scheduled_date  
FROM route;
```

2. View the name, userid, employeeid and role of the employees

**DDL Code:**

```
CREATE VIEW manager_view AS  
SELECT fname || ' ' || lname AS full_name,  
userid, employeeid, person_role  
FROM employee NATURAL JOIN user_login;
```

3. View all the available employees who are not assigned to any bus.

**DDL Code:**

```
CREATE VIEW available_employees AS
```

```
SELECT * FROM Employee NATURAL JOIN User_login  
WHERE busid IS null AND (person_role = 1 OR person_role = 2);
```

4. View the bookingid, mode of payment(offline or payment) for all the payments done

**DDL Code:**

```
CREATE VIEW payment_details AS  
SELECT bookingid, paymentid, payment_mode  
FROM payment;
```

5. View the id's of buses to which both driver and conductor are not assigned.

**DDL Code:**

```
CREATE VIEW bus_employee_not_assigned AS  
SELECT * FROM bus  
WHERE busid  
NOT IN (SELECT busid FROM employee WHERE busid IS NOT null);
```

**Stored procedures or Functions:**

1. Function to print the amount paid, paymentid of the customer (Takes the customerid as input).

**SQL Code:**

```
CREATE OR REPLACE FUNCTION amount_details(cid numeric(7))  
RETURNS numeric(8,4) AS $BODY$
```

DECLARE

amount numeric(8,4);

pid numeric(7);

BEGIN

SELECT amount\_paid,paymentid INTO amount, pid FROM  
customer NATURAL JOIN payment WHERE customerid = cid;

IF NOT FOUND THEN

RAISE EXCEPTION 'Customer with id % not found',cid;

END IF;

RAISE NOTICE 'Customer id = % ',cid;


RAISE NOTICE 'Payment id = % ',pid;

RETURN amount;

END;

\$BODY\$ LANGUAGE 'plpgsql';

## Output:

| Data Output   | Explain | Messages | Notifications |
|---|---------|----------|---------------|
|  amount_details<br>numeric |         |          |               |
| 1   |         | 500.0000 |               |

| Data Output                                     | Explain | Messages | Notifications |
|---|---------|----------|---------------|
| NOTICE: Customer id = 1347164                   |         |          |               |
| NOTICE: Payment id = 6153117                    |         |          |               |
| Successfully run. Total query runtime: 40 msec. |         |          |               |
| 1 rows affected.                                |         |          |               |

```

66     RETURN amount;
67
68 END;
69 $BODY$ LANGUAGE 'plpgsql';
70
71 SELECT amount_details(1347170);
72

```

Data Output Explain Messages Notifications

```

ERROR:  Customer with id 1347170 not found
CONTEXT:  PL/pgSQL function amount_details(numeric) line 10 at RAISE
SQL state: P0001

```

## 2. Function to count the number of customers.

### SQL Code:

```
CREATE OR REPLACE FUNCTION tot_customers()
```

```
RETURNS integer AS $BODY$
```

```
DECLARE
```

```
    total integer;
```

```
BEGIN
```

```
    SELECT COUNT(*) INTO total FROM customer;
```

```
    RETURN total;
```

```
END;
```

```
$BODY$ LANGUAGE 'plpgsql';
```

### Output:

88 SELECT tot\_customers();

89

Data Output

Explain

Messages

Notifications

|   | tot_customers<br>integer |  |
|---|--------------------------|--|
| 1 | 5                        |  |

- Function to display the name, userid and employeeid of the customers who belong to a particular city. (Takes the name of city as input).

### SQL Code:

```
CREATE OR REPLACE FUNCTION cus_details(place varchar(30))
RETURNS table(firstname varchar,lastname varchar, eid
numeric,uid numeric) AS $BODY$

BEGIN

    RETURN query

        SELECT fname :: varchar,lname :: varchar,employeeid ::
numeric, userid :: numeric FROM employee NATURAL JOIN
user_login

        WHERE district = place;

END;

$BODY$ LANGUAGE 'plpgsql';
```

### Output:

105SELECT \* from cus\_details('Mumbai');

106

Data Output

Explain

Messages

Notifications

|   | <div>firstname</div> <div>character varying</div> | <div>lastname</div> <div>character varying</div> | <div>eid</div> <div>numeric</div> | <div>uid</div> <div>numeric</div> |  |
|---|---|--|-----------------------------------|-----------------------------------|--|
| 1 | Sam   | Johnson  | 3167191                           | 1242121                           |  |
| 2 | Dev   | Kapoor   | 3167192                           | 1242122                           |  |
| 3 | Mare  | Jenner   | 3167193                           | 1242123                           |  |
| 4 | Kylie   | Jackson  | 3167194                           | 1242124                           |  |

## Triggers:

1. Trigger to update the seats available in a bus if a booking is deleted or inserted.

## SQL Code:

```
CREATE OR REPLACE FUNCTION process_seats_update()
```

```
RETURNS TRIGGER AS $seats_update$
```

```
BEGIN
```

```
    IF(tg_op = 'DELETE') THEN
```

```
        UPDATE bus SET available_number_of_seats =
available_number_of_seats + 1 WHERE busid = (SELECT busid FROM
seat NATURAL JOIN bookingid WHERE bookingid = OLD.bookingid);
```

```
        return old;
```

```
    ELSEIF (tg_op = 'INSERT') THEN
```

```
        UPDATE bus SET available_number_of_seats =
available_number_of_seats - 1 WHERE busid = (SELECT busid FROM
seat NATURAL JOIN bookingid WHERE bookingid = OLD.bookingid);
```

```
RETURN NEW;
```

```
END IF;
```

```
RETURN null;
```

```
END $seats_update$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER seats_update
```

```
AFTER INSERT OR DELETE ON booking
```

```
FOR EACH ROW EXECUTE PROCEDURE process_seats_update();
```