Birthday Email Scheduler - Database Documentation

1. Commands Related to the Table creation and their working.

1.1. Employees Table.

1.1.1. *Creating the table.*

```
CREATE TABLE employees (
id BIGINT PRIMARY KEY,
birthday DATE NOT NULL,
email VARCHAR (255) UNIQUE NOT NULL,
VARCHAR (255) NOT NULL );
```

1.1.2. *Import Data from a File to the Table.*

```
COPY employees (id, birthday, email, name) FROM
'C:\Users\Manish\Documents\Emailscheduler progress\employees.csv' DELIMITER ','
CSV HEADER;
```

1.1.3. *Grant All Privileges on a Table*

```
GRANT SELECT, INSERT, UPDATE, DELETE ON employees TO user_manish;
```

1.1.4. *Adding a unique Constraint.*

```
ALTER TABLE employees
ADD CONSTRAINT unique_email UNIQUE (email);
```

1.2. Email_templates Table.

1.2.1. *Creating a Table*

```
CREATE TABLE email_templates (
id BIGINT PRIMARY KEY,
image_url VARCHAR(255) NOT NULL,
message_body VARCHAR (255) NOT NULL);
```

1.2.2. *Import Data from a File to the Table. (For predefined templates)*

```
COPY email_templates (id, image_url, message_body) FROM
'C:\Users\Manish\Documents\Emailscheduler progress\email_templates.csv'
DELIMITER ',' CSV HEADER;
```

1.3. <u>Email_logs Table.</u>

1.3.1. Creating a table.

```
CREATE TABLE email_logs (
 id BIGINT PRIMARY KEY,
employee_id BIGINT NOT NULL, status VARCHAR(255) NOT NULL,
timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
email_id VARCHAR(255) NOT NULL,
error VARCHAR(255),
CONSTRAINT fk_employee FOREIGN KEY (employee_id) REFERENCES
employees(id) );
```

1.3.2. Adding a Foreign Key Constraint.

```
ALTER TABLE email_logs
ADD CONSTRAINT fk_employee FOREIGN KEY (employee_id)
REFERENCES employees(id);
```

Birthday Email Scheduler - Database Documentation

2. Database Schema Design and Description.

2.1. Employees Table

- **Purpose:** Stores information about employees, including their birthdays and contact details.
- **Key Columns:**
    - **id:** Primary key, a unique identifier for each employee.
    - **birthday:** The employee’s date of birth.
    - **email:** The employee’s email address (unique for each employee).
    - **name:** The full name of the employee.

2.2. email_templates Table

- **Purpose:** Stores predefined email templates for sending birthday emails.
- **Key Columns:**
    - **id:** Primary key, a unique identifier for each email template.
    - **image_url:** URL for the image included in the email.
    - **message_body:** The content of the email, including birthday wishes.

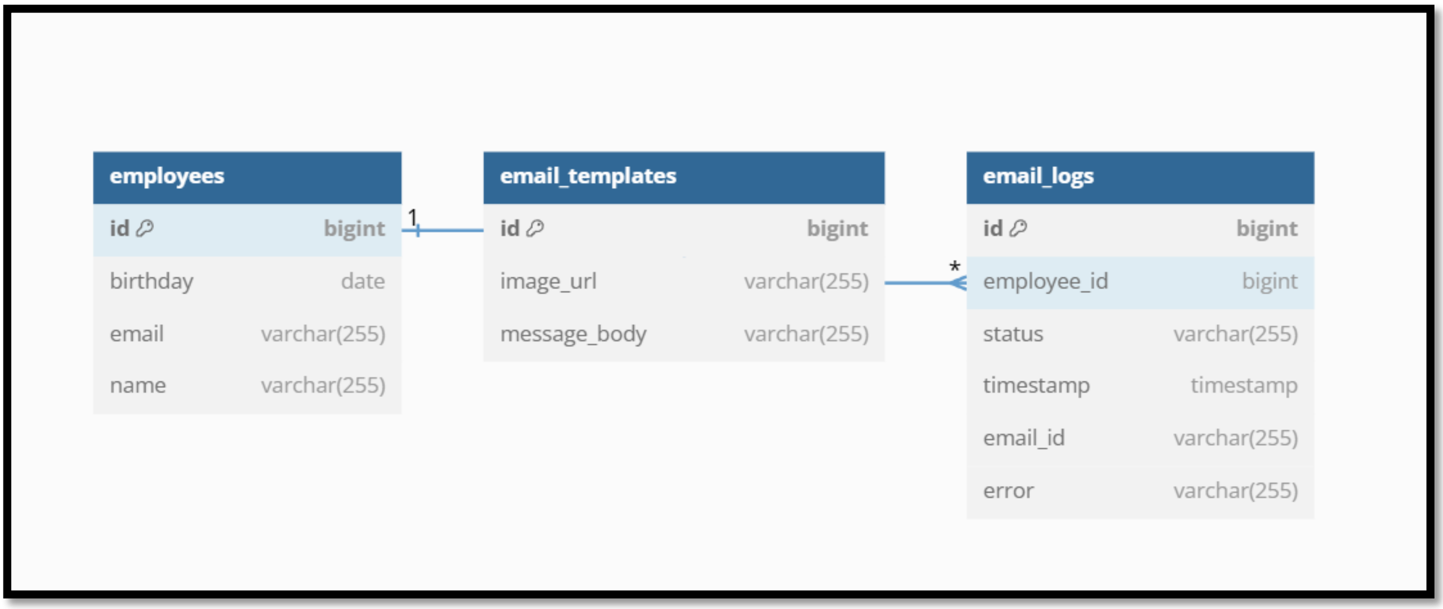
2.3. email_logs Table

- **Purpose:** Tracks the history and status of birthday emails sent to employees.
- **Key Columns:**
    - **id:** Primary key, a unique identifier for each email log entry.
    - **employee_id:** Foreign key referencing the id column in the employees table, linking each email log to an employee.
    - **status:** Indicates whether the email was successfully sent or failed.
    - **timestamp:** Records the date and time the email was sent.

- **email_id:** Stores the recipient's email address.
- **error:** Captures any error messages in case the email fails to send.

2.4. Relationships

1. **employees → email_logs:**
   - **Type:** One-to-Many.
   - **Explanation:** Each employee can have multiple email log entries, representing multiple emails sent over time.
2. **email_templates:**
   - Not explicitly linked but stores reusable templates for emails, referenced in the application logic when generating and sending emails.



employees	
id	bigint
birthday	date
email	varchar(255)
name	varchar(255)

email_templates	
id	bigint
image_url	varchar(255)
message_body	varchar(255)

email_logs	
id	bigint
employee_id	bigint
status	varchar(255)
timestamp	timestamp
email_id	varchar(255)
error	varchar(255)

Figure 1: Birthday_email_scheduler DB schema.