INSTALLING AND CONFIGURING POSTGRESQL AT UBUNTU

I am using postgres 9.6 for case sturdy and this will work for every other version in UBUNTU

Create the file repository configuration:

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'
```

Import the repository signing key:

```
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
```

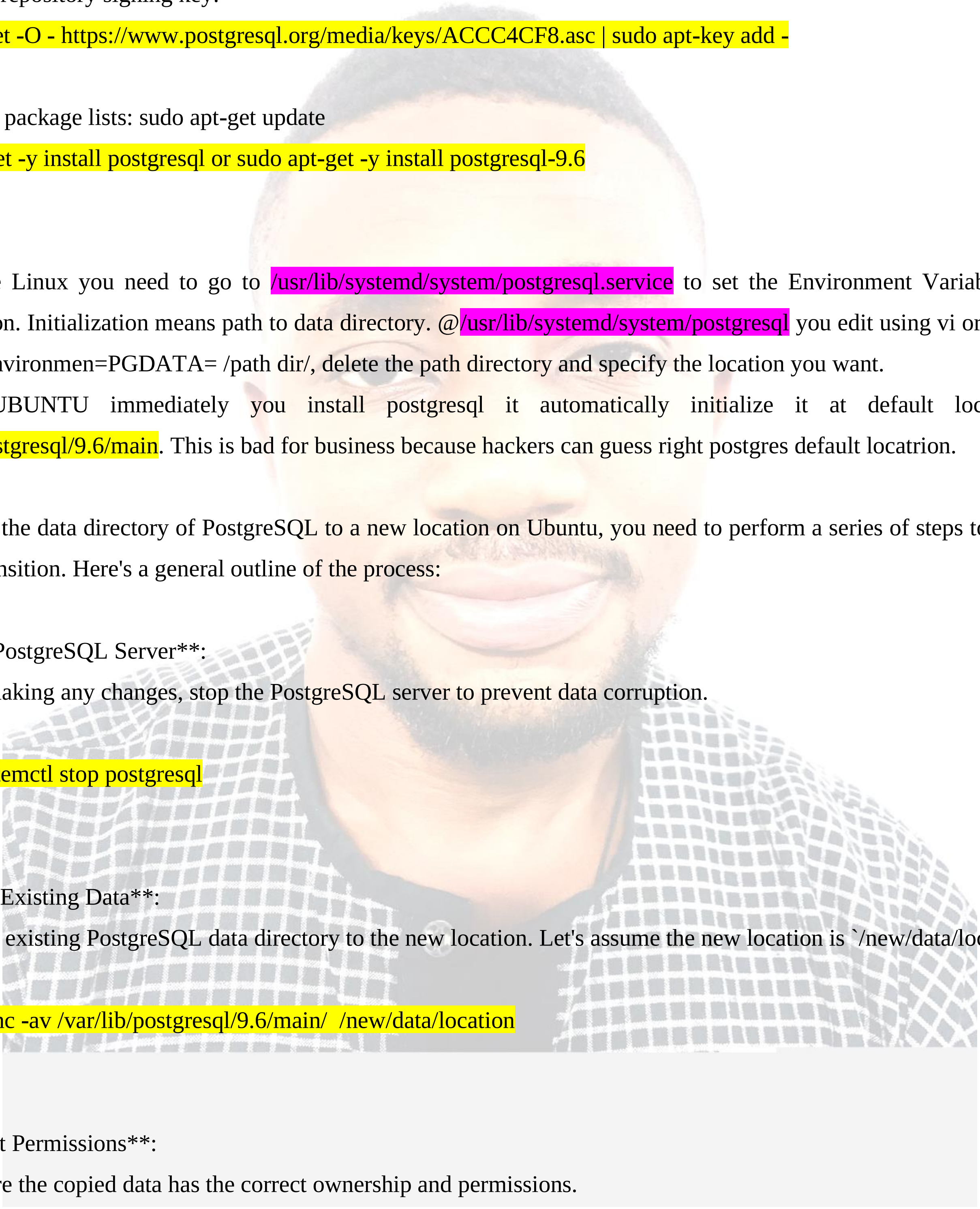
Update the package lists: sudo apt-get update

```
sudo apt-get -y install postgresql or sudo apt-get -y install postgresql-9.6
```

NOTE

For Oracle Linux you need to go to /usr/lib/systemd/system/postgresql.service to set the Environment Variable before initialization. Initialization means path to data directory. @/usr/lib/systemd/system/postgresql you edit using vi or nano and look for Environmen=PGDATA= /path dir/, delete the path directory and specify the location you want.

But in UBUNTU immediately you install postgresql it automatically initialize it at default location @ /var/lib/postgresql/9.6/main. This is bad for business because hackers can guess right postgres default locatrion.



To change the data directory of PostgreSQL to a new location on Ubuntu, you need to perform a series of steps to ensure a smooth transition. Here's a general outline of the process:

1. **Stop PostgreSQL Server**:

   Before making any changes, stop the PostgreSQL server to prevent data corruption.

   ```bash
   sudo systemctl stop postgresql
   ```

2. **Copy Existing Data**:

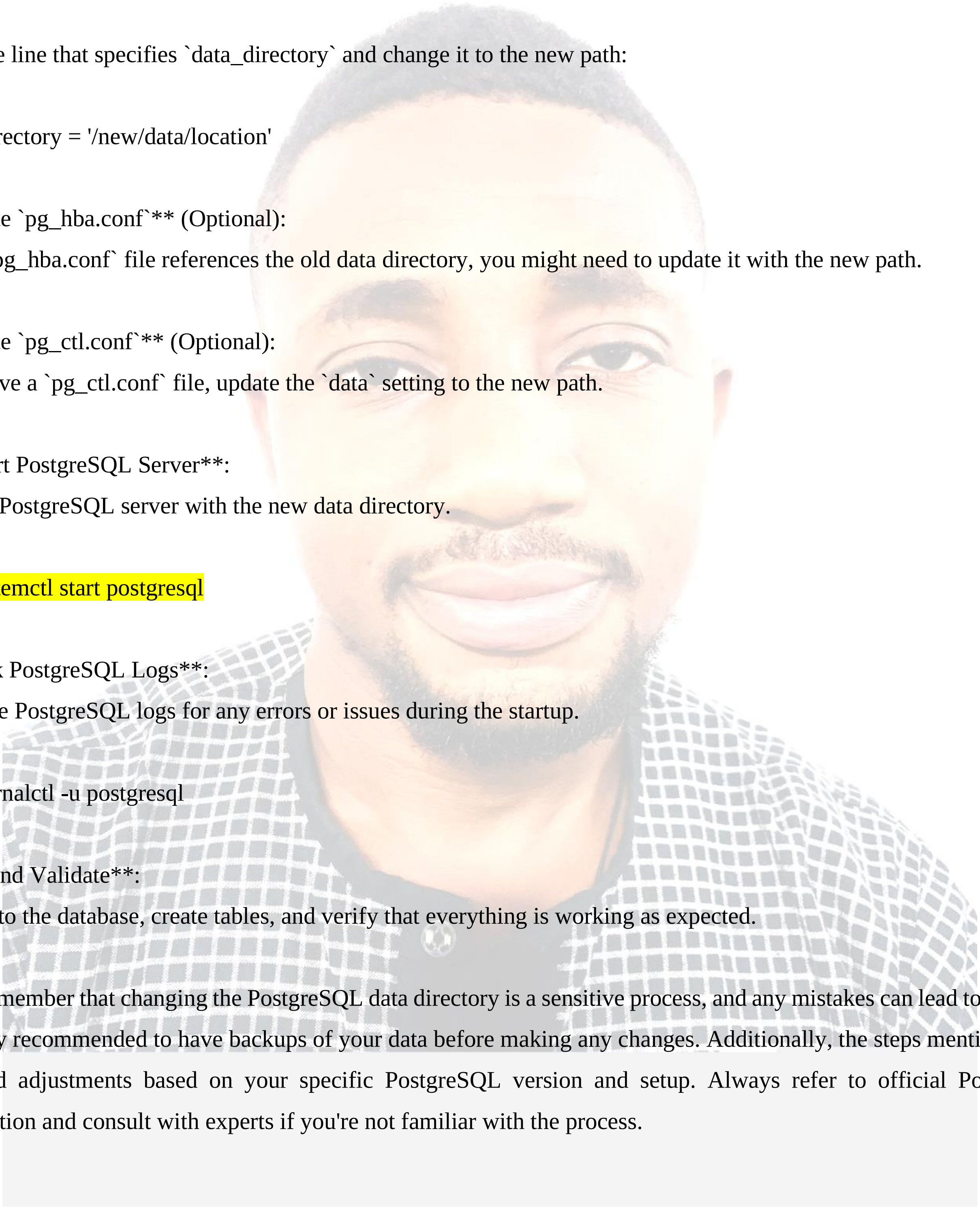
   Copy the existing PostgreSQL data directory to the new location. Let's assume the new location is `/new/data/location`.

   ```bash
   sudo rsync -av /var/lib/postgresql/9.6/main/  /new/data/location
   ```

3. **Adjust Permissions**:

   Make sure the copied data has the correct ownership and permissions.

   ```bash
   sudo chown -R postgres:postgres /new/data/location
   sudo chmod 700 /new/data/location
   ```



4. **Update Configuration Files**:

   Edit the PostgreSQL configuration files to point to the new data directory.

   - Open the PostgreSQL configuration file (`postgresql.conf`):

     ```bash
     sudo nano /etc/postgresql/{version}/main/postgresql.conf
     ```

   - Find the line that specifies `data_directory` and change it to the new path:

     ```
     data_directory = '/new/data/location'
     ```

5. **Update `pg_hba.conf`** (Optional):

   If your `pg_hba.conf` file references the old data directory, you might need to update it with the new path.

6. **Update `pg_ctl.conf`** (Optional):

   If you have a `pg_ctl.conf` file, update the `data` setting to the new path.

7. **Restart PostgreSQL Server**:

   Start the PostgreSQL server with the new data directory.

   ```bash
   sudo systemctl start postgresql
   ```

8. **Check PostgreSQL Logs**:

   Check the PostgreSQL logs for any errors or issues during the startup.

   ```bash
   sudo journalctl -u postgresql
   ```

9. **Test and Validate**:

   Connect to the database, create tables, and verify that everything is working as expected.

NOTE: Remember that changing the PostgreSQL data directory is a sensitive process, and any mistakes can lead to data loss. It's strongly recommended to have backups of your data before making any changes. Additionally, the steps mentioned here might need adjustments based on your specific PostgreSQL version and setup. Always refer to official PostgreSQL documentation and consult with experts if you're not familiar with the process.

DIFFRENCE BETWEEN POSTGRES AT ORACLE LINUX AND UBUNTU

@Oracle Linux the whole data directory is in one place specified after initialization

@Ubuntu It initialize automatically immediately after install which keeps the data directory at Default location except the postgresaql.conf and pg_hba.conf which it will keep at /etc/postgresql/9.6/main

WHAT RSYNC DOES

The `rsync` command with the `-av` options is used to synchronize files and directories between two locations. Here's what each option does:

- `-a` (archive mode): This option enables archive mode, which is a comprehensive way of copying files while preserving permissions, ownership, timestamps, symbolic links, and more. It's a shorthand for several options combined, including `-r` (recursive), `-l` (copy symlinks as symlinks), `-p` (preserve permissions), `-t` (preserve modification times), and more.

- `-v` (verbose): This option displays detailed information about the progress of the synchronization, showing each file that's being copied and any other relevant information.

When you use the command `rsync -av source destination`, you are synchronizing the contents of the `source` directory to the `destination` directory, while preserving all the attributes mentioned above. This is often used for tasks such as creating backups, transferring files between systems, or keeping two directories in sync.

For example:

```bash
rsync -av /path/to/source/ /path/to/destination/
```

This command will copy all the contents of the `source` directory to the `destination` directory, including subdirectories and their contents, while maintaining the permissions, ownership, timestamps, and other attributes. The `-v` option will display detailed progress information in the terminal.

Remember to include the trailing slashes (`/`) in the paths to ensure that the contents of the source directory are copied into the destination directory rather than creating a subdirectory with the source directory's name within the destination directory.