How to execute an incremental backup in PostgreSQL?

Why is an incremental backup necessary?

Incremental backups save only the changes made since the last backup, instead of backing up the entire database each time. This reduces storage needs and backup time, especially for large databases with small changes.

What is the backup_manifest file in an incremental backup?

A **backup_manifest** is a file that contains metadata about a backup, describing the contents, structure, and important information needed to restore the backup. It acts like a catalog or inventory of the backup, ensuring that all necessary data is properly tracked and available for recovery.

What is pg_combinebackup?

To combine all the incremental backups, use the **pg_combinebackup** utility. It is introduced in PostgreSQL v17, it reconstructs a full backup from incremental backups.

The user needs to follow these steps to perform an incremental backup in PostgreSQL v17.

1. Install PostgreSQL v17 and enable this configuration.

```
summarize_wal = 'on'
```

2. Create a database and then create the following table with the specified data.

```
create database incr_backup ;
CREATE DATABASE

INSERT INTO employees (name, position, salary)
VALUES
    ('Alice', 'Manager', 75000),
    ('Bob', 'Developer', 55000),
    ('Charlie', 'Analyst', 50000);
```

3. First, create a full backup of the database and use it as the baseline for incremental backups.

```
 pg_basebackup -D /var/lib/pgsql/17/full_backup/ -c fast -p 5432 -v
```

```
pg_basebackup: initiating base backup, waiting for checkpoint to complete
pg_basebackup: checkpoint completed
pg_basebackup: write-ahead log start point: 0/F000028 on timeline 1
pg_basebackup: starting background WAL receiver
pg_basebackup: created temporary replication slot "pg_basebackup_357"
pg_basebackup: write-ahead log end point: 0/F000120
pg_basebackup: waiting for background process to finish streaming ...
pg_basebackup: syncing data to disk ...
pg_basebackup: renaming backup_manifest.tmp to backup_manifest
pg_basebackup: base backup completed
```

4. Insert additional data and create the first incremental backup in the series.

```
 INSERT INTO employees (name, position, salary)
VALUES ('Diana', 'Designer', 60000);
```

Note: This utilizes the backup_manifest from the full backup to capture only the changes made since that backup.

```
pg_basebackup -D /var/lib/pgsql/17/employees_db_incremental_1stbackup -i
/var/lib/pgsql/17/full_backup/backup_manifest -c fast -p 5432 -v
```

```
pg_basebackup: initiating base backup, waiting for checkpoint to complete
pg_basebackup: checkpoint completed
pg_basebackup: write-ahead log start point: 0/11000028 on timeline 1
pg_basebackup: starting background WAL receiver
pg_basebackup: created temporary replication slot "pg_basebackup_370"
pg_basebackup: write-ahead log end point: 0/11000120
pg_basebackup: waiting for background process to finish streaming ...
pg_basebackup: syncing data to disk ...
pg_basebackup: renaming backup_manifest.tmp to backup_manifest
pg_basebackup: base backup completed
```

5. Insert more data and create the second incremental backup in the series.

```
INSERT INTO employees (name, position, salary)
```

```
VALUES ('Alex', 'Sportsman', 90000);
```

Note: This uses the backup_manifest from the first backup to capture only the changes made since that backup.

```
pg_basebackup -D /var/lib/pgsql/17/employees_db_incremental_2ndbackup -i
/var/lib/pgsql/17/employees_db_incre
mental_1stbackup/backup_manifest -c fast -p 5432 -v
```

```
pg_basebackup: initiating base backup, waiting for checkpoint to complete
pg_basebackup: checkpoint completed
pg_basebackup: write-ahead log start point: 0/13000028 on timeline 1
pg_basebackup: starting background WAL receiver
pg_basebackup: created temporary replication slot "pg_basebackup_385"
pg_basebackup: write-ahead log end point: 0/13000120
pg_basebackup: waiting for background process to finish streaming ...
pg_basebackup: syncing data to disk ...
pg_basebackup: renaming backup_manifest.tmp to backup_manifest
pg_basebackup: base backup completed
```

6. Insert additional data and create the third incremental backup in the series.

```
INSERT INTO employees (name, position, salary)
VALUES ('Messi', 'Athelete', 90000);
```

Note: This uses the backup_manifest from the second backup to capture only the changes made since that backup.

```
pg_basebackup -D /var/lib/pgsql/17/employees_db_incremental_3rdbackup -i
/var/lib/pgsql/17/employees_db_incremental_2ndbackup/backup_manifest -c
fast -p 5432 -v
```

```
pg_basebackup: initiating base backup, waiting for checkpoint to complete
pg_basebackup: checkpoint completed
pg_basebackup: write-ahead log start point: 0/15000028 on timeline 1
pg_basebackup: starting background WAL receiver
pg_basebackup: created temporary replication slot "pg_basebackup_395"
```

```
pg_basebackup: write-ahead log end point: 0/15000120
pg_basebackup: waiting for background process to finish streaming ...
pg_basebackup: syncing data to disk ...
pg_basebackup: renaming backup_manifest.tmp to backup_manifest
pg_basebackup: base backup completed
```

7. To combine all the backups, use the **pg_combinebackup** utility.

```
 pg_combinebackup -o /var/lib/pgsql/17/combined_backup
/var/lib/pgsql/17/full_backup /var/lib/pgsql/17/employees_db_incremental_
1stbackup /var/lib/pgsql/17/employees_db_incremental_2ndbackup
/var/lib/pgsql/17/employees_db_incremental_3rdbackup
```

8. Edit the port postgresql.conf in combined backup and start the server

```
vi /var/lib/pgsql/17/combined_backup/postgresql.conf
```

```
./pg_ctl -D /var/lib/pgsql/17/combined_backup/ start
waiting for server to start....2024-11-20 17:36:41.958 UTC [413] LOG:
redirecting log output to logging collector process
2024-11-20 17:36:41.958 UTC [413] HINT:  Future log output will appear in
directory "log".
 done
server started
```

```
[postgres@dfe4dd455d96 bin]$ psql -U postgres incr_backup -p 6432
psql (17.0)
Type "help" for help.

incr_backup=#
incr_backup=# select * from employees;
 employee_id |  name   | position  |  salary  |         updated_at
-------------+---------+-----------+----------+----------------------------
           1 | Alice   | Manager   | 75000.00 | 2024-11-20 17:25:08.625332
           2 | Bob     | Developer | 55000.00 | 2024-11-20 17:25:08.625332
           3 | Charlie | Analyst   | 50000.00 | 2024-11-20 17:25:08.625332
           4 | Diana   | Designer  | 60000.00 | 2024-11-20 17:27:22.876933
           5 | Alex    | Sportsman | 90000.00 | 2024-11-20 17:28:57.633128
```

```
          6 | Messi    | Athelete   | 90000.00 | 2024-11-20 17:30:58.751833
(6 rows)
```