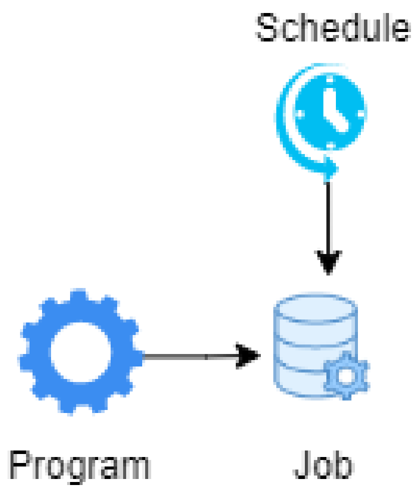
DBMS Program Job & Scheduler

- DBMS_SCHEDULER is package schedules and manages jobs in the job queue.
- **Program** is all about the metadata of a task without scheduling information and can be created on PLSQL Block, Store Procedure, Shell Executable. Program can have arguments also.
- **Schedule** is all about the frequency at which the Job will be run. It may optionally have Start Time and End Time
- **Job** is nothing but the execution of program at predefined schedule. Basically Job can have inline program and schedule as well as can be created based on predefined existing program and schedule.



Create a Simple Program

```
BEGIN
 DBMS_SCHEDULER.create_program (
  program_name   => ‘PROG_NAME',
  program_type   => 'PLSQL_BLOCK',
  program_action => 'BEGIN sample_pkg.sample_proc; END;',
  enabled        => TRUE,
  comments       => 'Sample Program Description');
END;
```

Display the Program Details

```
SELECT owner, program_name, enabled FROM dba_scheduler_programs;
```

Drop the Program

```
BEGIN
 DBMS_SCHEDULER.drop_program (program_name => 'PROG_NAME');
END;
```

DBMS Program Job & Scheduler

Create a Program with Program Arguments

```
BEGIN
 DBMS_SCHEDULER.create_program (
  program_name       => 'PROG_NAME',
  program_type       => 'STORED_PROCEDURE',
  program_action     => 'sample_pkg.sample_proc',
  number_of_arguments => 1,
  enabled            => FALSE,
  comments           => 'Sample Program Description');

 DBMS_SCHEDULER.DEFINE_PROGRAM_ARGUMENT (
  program_name      => 'PROG_NAME',
  argument_name     => 'PARAM1',
  argument_position => 1,
  argument_type     => 'VARCHAR2',
  default_value     => 'Value1');

 DBMS_SCHEDULER.ENABLE (name => ‘PROG_NAME');
END;
```

Enable & Disable Program

```
BEGIN
 DBMS_SCHEDULER.ENABLE(name => 'PROG_NAME');
 DBMS_SCHEDULER.DISABLE(name => 'PROG_NAME');
END;
```

DBMS Program Job & Scheduler

Create a Job (Inline Program & Inline Schedule)

```
BEGIN
 DBMS_SCHEDULER.CREATE_JOB (
  job_name      => 'JOB_NAME',
  job_type      => 'PLSQL_BLOCK',
  job_action    => 'BEGIN sample_pkg.sample_proc; END;',
  start_date    => SYSTIMESTAMP,
  repeat_interval => 'FREQ=HOURLY; INTERVAL=12;',
  enabled       => TRUE);
END;
```

Create a Job based on Program with Inline Schedule

```
BEGIN
 DBMS_SCHEDULER.CREATE_JOB (
  job_name      => 'JOB_NAME',
  program_name  => 'PROG_NAME',
  start_date    => SYSTIMESTAMP,
  repeat_interval => 'FREQ=HOURLY; INTERVAL=12;',
  end_date      => NULL,
  enabled       => TRUE,
  comments      => 'Sample Job');
END;
```

Display the Job Details

```
SELECT owner, job_name, enabled FROM dba_scheduler_jobs;
```

Drop the Job

```
BEGIN
 DBMS_SCHEDULER.DROP_JOB (job_name => 'JOB_NAME');
END;
```

Enable & Disable Job

```
BEGIN
 DBMS_SCHEDULER.ENABLE(name => 'JOB_NAME');
 DBMS_SCHEDULER.DISABLE(name => 'JOB_NAME');
END;
```

Change/ Set Job Attributes

```
BEGIN
 DBMS_SCHEDULER.SET_ATTRIBUTE (
  name      => 'CUSTOM_SCHEDULE',
  attribute => 'REPEAT_INTERVAL',
  value     => 'FREQ=HOURLY;');
END;
```

DBMS Program Job & Scheduler

Create a Schedule

```
BEGIN
  DBMS_SCHEDULER.CREATE_SCHEDULE(
  schedule_name  => ‘CUSTOM_SCHEDULE',
  start_date     => trunc(sysdate)+18/24,
  repeat_interval => 'freq=HOURLY;interval=1',
  comments       => 'Runtime: Every day every hour'
);
END;
```

Create a Job based on Existing Program & Schedule

```
BEGIN
  DBMS_SCHEDULER.create_job (
    job_name      => 'JOB_NAME',
    program_name  => 'PROG_NAME',
    schedule_name => 'CUSTOM_SCHEDULE',
    enabled       => TRUE,
    comments      => 'Sample Job Description');
END;
```

Display the Schedule Details

```
SELECT owner, schedule_name FROM dba_scheduler_schedules;
```

Drop the Schedule

```
BEGIN
  DBMS_SCHEDULER.DROP_SCHEDULE (
        schedule_name => 'CUSTOM_SCHEDULE‘
);
END;
```

Run Job Synchronously

```
BEGIN
  DBMS_SCHEDULER.run_job (
    job_name            => 'JOB_NAME',
    use_current_session => TRUE);
END;
```

Stop Job

```
BEGIN
  DBMS_SCHEDULER.stop_job (job_name => 'JOB_NAME');
END;
```