```
DROP EVENT IF EXISTS business_events.jobs_polling_event_scheduler;
DROP PROCEDURE IF EXISTS business_events.transferJobs;

CREATE PROCEDURE business_events.transferJobs()
  BEGIN
    DECLARE exit handler for sqlexception
    BEGIN
      ROLLBACK;
    END;

    START TRANSACTION;
      PREPARE stmt1 FROM 'UPDATE ScheduledBusinessEvents SET status =
"InProgress" WHERE executionTime <= NOW() AND status IS NULL';
      EXECUTE stmt1;
      PREPARE stmt2 FROM 'INSERT IGNORE INTO
business_events.ScheduledBusinessEventsExecution(scheduledBusinessEvent
Key,clientId,eventId,businessEventId,entityId,executionTime,systemActor,schem
aEntityMapping,context,sorTriggerType,dateCreated,lastModified,mask)
SELECT
scheduledBusinessEventKey,clientId,eventId,businessEventId,entityId,execution
Time,systemActor,schemaEntityMapping,context,sorTriggerType,dateCreated,la
stModified,mask FROM ScheduledBusinessEvents WHERE status =
"InProgress"';
      EXECUTE stmt2;
      PREPARE stmt3 FROM 'UPDATE ScheduledBusinessEvents SET status =
"Completed" WHERE executionTime <= NOW() AND status = "InProgress"';
      EXECUTE stmt3;
    COMMIT;
  END;


CREATE EVENT IF NOT EXISTS business_events.jobs_polling_event_scheduler
  ON SCHEDULE EVERY 20 SECOND
  ON COMPLETION PRESERVE
  DO
  BEGIN
    CALL transferJobs();
  END;



  show events;
  SET GLOBAL event_scheduler = ON;

  truncate table ScheduledBusinessEvents;

  DROP PROCEDURE IF EXISTS transferJobs;
DELIMITER $$
 CREATE PROCEDURE transferJobs()
 BEGIN

 DECLARE exit handler for sqlexception
   BEGIN
   ROLLBACK;
 END;

 START TRANSACTION;
PREPARE stmt1 FROM 'UPDATE ScheduledBusinessEvents SET status =
"InProgress" WHERE executionTime <= NOW() AND status IS NULL';
      EXECUTE stmt1;
      PREPARE stmt2 FROM 'INSERT IGNORE INTO
business_events.ScheduledBusinessEventsExecution(scheduledBusinessEvent
Key,clientId,eventId,businessEventId,entityId,executionTime,systemActor,schem
aEntityMapping,context,sorTriggerType,dateCreated,lastModified,mask)
SELECT
scheduledBusinessEventKey,clientId,eventId,businessEventId,entityId,execution
Time,systemActor,schemaEntityMapping,context,sorTriggerType,dateCreated,la
stModified,mask FROM ScheduledBusinessEvents WHERE status =
"InProgress"';
      EXECUTE stmt2;
      PREPARE stmt3 FROM 'UPDATE ScheduledBusinessEvents SET status =
"Completed" WHERE executionTime <= NOW() AND status = "InProgress"';
      EXECUTE stmt3;
COMMIT;
 END
 $$
 DELIMITER ;
```

```
DROP EVENT IF EXISTS jobs_polling_event_scheduler;
DELIMITER $$

CREATE EVENT IF NOT EXISTS jobs_polling_event_scheduler
ON SCHEDULE EVERY 20 SECOND
ON COMPLETION PRESERVE
DO
BEGIN

CALL transferJobs();

END$$
DELIMITER ;
```