1. Store procedure for our first advanced query

```
CREATE DEFINER=`root`@`%` PROCEDURE `GetMostFlightDelayByAirline`()
BEGIN
    DECLARE varAirline VARCHAR(50);
    DECLARE varmostDelays LONG;
    DECLARE message VARCHAR(100);

    DECLARE exit_loop BOOLEAN DEFAULT FALSE;
    DECLARE custCur CURSOR FOR
        (SELECT A.name, COUNT(Ac.airlineCode) AS mostDelayss
        FROM Airline A JOIN FlightActual Ac ON A.airlineCode = Ac.airlineCode
        WHERE Ac.arrivalDelayTime > 0
        GROUP BY A.airlineCode
        ORDER BY mostDelayss DESC
        );

    DECLARE CONTINUE HANDLER FOR NOT FOUND SET exit_loop = TRUE;

    DROP TABLE IF EXISTS NewTable;

    CREATE TABLE NewTable (
      airlineName    VARCHAR(100) Primary Key,
      mostDelays INT,
      message2 VARCHAR(100)
    );

    OPEN custCur;
        cloop: LOOP
        FETCH custCur INTO varAirline, varmostDelays;
        IF exit_loop THEN
            LEAVE cloop;
        END IF;

        IF varmostDelays > 687.0 THEN
            SET message = "Only pick this airline if you have to!!!!
[Avoid!]";
        ELSEIF varmostDelays > 300.0 THEN
            SET message = "There are better airlines...";
        ELSE
            SET message = "Best airline to travel!";
        END IF;

        INSERT INTO NewTable VALUE (varAirline, varmostDelays, message);
        END LOOP cloop;
     CLOSE custCur;

     SELECT airlineName, mostDelays, message2
     FROM NewTable
     ORDER BY airlineName;
END
```

2. Store procedure for our second advanced query

```
      CREATE DEFINER=`root`@`%` PROCEDURE `GetStateDelayTime`()
      BEGIN
          DECLARE varState VARCHAR(2);
          DECLARE varAvg_delay LONG;
          DECLARE varMajor_delay VARCHAR(100);

          DECLARE exit_loop BOOLEAN DEFAULT FALSE;
          DECLARE custCur CURSOR FOR
              (SELECT A.state AS state, AVG(Ac.arrivalDelayTime) AS avg_delay
              FROM FlightActual Ac JOIN FlightSchedule F ON Ac.flightNumber =
F.flightNumber AND Ac.date = F.date AND Ac.tailNumber = F.tailNumber JOIN
Airport A ON F.airportDestinationCode = A.airportCode
              WHERE Ac.arrivalDelayTime > 0
              GROUP BY A.state
              ORDER BY A.state);

          DECLARE CONTINUE HANDLER FOR NOT FOUND SET exit_loop = TRUE;

          DROP TABLE IF EXISTS NewTable;

          CREATE TABLE NewTable (
            state    VARCHAR(100) Primary Key,
            avg_delay LONG,
            major_Delay VARCHAR(100)
          );

          OPEN custCur;
              cloop: LOOP
              FETCH custCur INTO varState, varAvg_delay;
              IF exit_loop THEN
                  LEAVE cloop;
              END IF;

              IF varAvg_delay > 30 THEN
                  SET varMajor_delay = "MAJOR delay";
              ELSEIF varAvg_delay > 20 THEN
                  SET varMajor_delay = "Some delay";
              ELSE
                  SET varMajor_delay = "Little to no delay";
              END IF;

              INSERT INTO NewTable VALUE (varState, varAvg_delay,
varMajor_delay);
              END LOOP cloop;
           CLOSE custCur;

           SELECT state, avg_delay, major_Delay
           FROM NewTable;
      END
```

Trigger

```
CREATE DEFINER=`root`@`%` TRIGGER `FlightSchedule_BEFORE_INSERT` BEFORE
INSERT ON `FlightSchedule` FOR EACH ROW BEGIN
    DECLARE num_delays INT;
    SELECT COUNT(*) INTO num_delays FROM DelayedFlights
    WHERE flightNumber = NEW.flightNumber
    AND date = NEW.date
    AND tailNumber = NEW.tailNumber
    AND scheduledArrival = NEW.scheduledArrival
    AND airlineCode = NEW.airlineCode;

    IF num_delays <> 0 THEN
        DELETE FROM DelayedFlights WHERE flightNumber = NEW.flightNumber AND
date = NEW.date AND tailNumber = NEW.tailNumber AND scheduledArrival =
NEW.scheduledArrival AND airlineCode = NEW.airlineCode;
    END IF;
END
```