A Queries used for paper in order of appearance

A.1 7.1 Count of all records

MySQL:

```
SELECT
    COUNT(speed)
FROM
    can_decode_message_speed
WHERE
    systime > 1668405600
    AND systime < 1668492000;
```

58744420 in 25.39s

IoTDB:

```
SELECT
    COUNT(speed\_speed)
FROM
    root.CIRCLES.*.2022\_11\_14.*.can\_bus\_data
GROUP BY LEVEL = 3
```

60475996 unique records 0.054s

A.2 Corrected MySQL count of all records

MySQL:

```
SELECT
    COUNT(*)
FROM
    can\_decode\_message\_speed
WHERE
    file\_tag\_id LIKE '%221114%'
```

60475996 in 13m 53.12s

A.3 7.2 Complexity (Fig. 2)

MySQL:

```
SELECT
    speed_intervals.time_interval,
    speed_intervals.speed AS last_speed,
    acc_intervals.set_speed AS last_set_speed
FROM
    (SELECT
         FROM_UNIXTIME(FLOOR(systime / 120) * 120) AS time_interval,
         speed,
         ROW_NUMBER() OVER (PARTITION BY FLOOR(systime / 120)
         ORDER BY systime DESC) AS rn
     FROM
         can_decode_message_speed
     WHERE
         systime > 1668747600 AND systime < 1668837600
         AND vin_key = 83
    ) AS speed_intervals
LEFT JOIN
    (SELECT
         FROM_UNIXTIME(FLOOR(systime / 120) * 120) AS time_interval,
         set_speed,
         ROW_NUMBER() OVER (PARTITION BY FLOOR(systime / 120)
         ORDER BY systime DESC) AS rn
     FROM
         can_decode_message_acc_hub
     WHERE
         systime > 1668747600 AND systime < 1668837600
         AND vin_key = 83
    ) AS acc_intervals
ON
    speed_intervals.time_interval = acc_intervals.time_interval AND
    speed_intervals.rn = acc_intervals.rn
WHERE
    speed_intervals.rn = 1
ORDER BY
    speed_intervals.time_interval;
```

100 rows in set (4 min 50.64 sec)

IoTDB:

```
SELECT
    LAST_VALUE(acc_hub_set_speed),
    LAST_VALUE(speed_speed)
FROM root.CIRCLES.vehicle_83.2022_11_18.*.can_bus_data
GROUP BY
    ([2022-11-18T00:00:00, 2022-11-18T23:59:59), 2m)
```

Total line number = 720 It costs 0.722s

A.4 7.3 Reinventing the Wheel: Calculation

IoTDB:

```
SELECT LAST_VALUE(acc_hub_set_speed) as set_speed,
LAST_VALUE(speed_speed) as speed,
LAST_VALUE(Lat) as Latitude,
LAST_VALUE(Long) as Longitude,
ABS(
    DEGREES(atan(
        ABS(LAST_VALUE(Long) - FIRST_VALUE(Long))
        /
        ABS(LAST_VALUE(Lat) - FIRST_VALUE(Lat))
    ))
    +jexl(
        (LAST_VALUE(Long)-FIRST_VALUE(Long)),
        (LAST_VALUE(Lat)-FIRST_VALUE(Lat)),
        'expr'='(x,y) -> (x < 0 && y >= 0 ? -360 : (x < 0 && y < 0 ? 180 : (x > 0 && y < 0 ? -180 : 0)))'
    )
) as Bearing
FROM root.CIRCLES.vehicle_83.2022_11_18.run_1.*
GROUP BY ([2022-11-18T07:30:00, 2022-11-18T09:30:00), 30s)
```

Total line number = 240 It costs 0.162s

A.5 Fig. 3 and 4, Cruise vs actual speed

IoTDB:

```
SELECT
    AVG(acc_hub_set_speed) - AVG(speed_speed)*0.621371
FROM
    root.circles100.vehicle_8*.2022_11_17.*
WHERE
    cruise_state_cruise_state > 0
    AND acc_hub_set_speed BETWEEN 0 AND 120
GROUP BY ([2022-11-17T07:00:00,2022-11-17T08:59:59), 10s)
ORDER BY Time
ALIGN BY DEVICE
```

23760 records in 0.3319981669774279 seconds dataframe conversion in 0.22297350002918392 seconds

A.6 Fig. 5 and 6, Jerk

IoTDB:

```
SELECT
    derivative(
        derivative(
            AVG (speed_speed))) AS jerk
FROM
    root.circles100.*.2022_11_17.*
GROUP BY ([2022-11-17T07:00:00,2022-11-17T08:59:59), 10s)
ALIGN BY DEVICE
```

216720 records in 0.32244883303064853 seconds dataframe conversion in 2.3397924170130864 seconds

A.7 Fig. 7 Space vs Time

IoTDB:

```
SELECT
    AVG(gps_data_Lat) AS Lat,
    AVG(gps_data_Long) AS Long,
    AVG(speed_speed)*0.621371 as Speed,
ABS(
    DEGREES(atan(
        ABS(LAST_VALUE(gps_data_Long) - FIRST_VALUE(gps_data_Long))
```

```
        /
        ABS(LAST_VALUE(gps_data_Lat) - FIRST_VALUE(gps_data_Lat))
    ))
    +jexl(
        (LAST_VALUE(gps_data_Long) - FIRST_VALUE(gps_data_Long)),
        (LAST_VALUE(gps_data_Lat) - FIRST_VALUE(gps_data_Lat)),
        'expr'='(x,y) -> (x < 0 && y >= 0 ? -360 : (x < 0 && y < 0 ? 180 : (x > 0 && y < 0 ? -180 : 0)))'
    )
) as Bearing
FROM root.circles100.*.2022_11_17.*
 group by ([2022-11-17T07:00:00,2022-11-17T08:59:59), 10s)  align by device
```

239040 records in 0.33727529196767136 seconds dataframe conversion in 3.454556083015632 seconds