Aum Amriteshwaryai Namah

Ramya Mercy Rajan- AA.SC.P2MCA2107434

Lab Assignment 3

-- 1. Create a keyspace for the application. Keyspace name: `tranz`.

```
CREATE KEYSPACE IF NOT EXISTS tranz with replication =
{'class': 'NetworkTopologyStrategy', 'datacenter1': 6};
        USE tranz;
```

-- 2.Create a table for drivers if not exists. Table name: `driver`. Columns: `driver_name` (unique, if not exists, primary key), `password` (string), `mobile` (number), `current_position` (string), `skill` (`set` type with strings).

```
        CREATE TABLE IF NOT EXISTS tranz.driver (
          driver_name TEXT,    password TEXT,
          mobile INT,       current_position TEXT,
          skill SET<TEXT>,      PRIMARY KEY (driver_name)   );
```

-- 3.Create index We will filter drivers by skill ,password and current_position. So create index on both

```
        CREATE INDEX IF NOT EXISTS driver_skill_index ON
tranz.driver (skill);
```

CREATE INDEX IF NOT EXISTS driver_current_position_index ON tranz.driver (current_position);

CREATE INDEX driver_password_index ON tranz.driver (password);

```
cqlsh:tranz> CREATE INDEX IF NOT EXISTS driver_skill_index ON tranz.driver (skill);
Bad Request: Indexes on collections are no yet supported
cqlsh:tranz> CREATE INDEX IF NOT EXISTS driver_current_position_index ON tranz.driver
         ... (current_position);
cqlsh:tranz> CREATE INDEX driver_password_index ON tranz.driver (password);
```

-- 4 Create a table for vehicles. Table name: `vehicle`. Columns: `vehicle_id` (string, unique, if not exists), `status` (string), type (string)

CREATE TABLE IF NOT EXISTS tranz.vehicle (
vehicle_id TEXT, status TEXT,
type TEXT, PRIMARY KEY (vehicle_id));

```
cqlsh:tranz> CREATE TABLE IF NOT EXISTS tranz.vehicle (
         ...  vehicle_id TEXT, status TEXT,
         ...  type TEXT, PRIMARY KEY (vehicle_id));
```

--5. create index on status

CREATE INDEX vehicle_status_index on tranz.vehicle (status);

```
cqlsh:tranz> CREATE INDEX vehicle_status_index on tranz.vehicle (status);
```

-- 6. Create a table for timetables. Table name: `time_table`. Columns: `line_name` (unique, if not exists, string), `service_no` (number, asc within line_name), `station_name` (string), `latitude` (double), `longitude` (double), `time` (int), `distance` (double), Notes: time are departure times, except the last (destination) time, it is arrival time. Sorted `asc` by `time`.

CREATE TABLE IF NOT EXISTS tranz.time_table (
line_name TEXT,
service_no INT,
station_name TEXT,

```
  longitude DOUBLE,
  latitude DOUBLE,
  time INT,
  distance DOUBLE,
  PRIMARY KEY (line_name, service_no, time)
) WITH CLUSTERING ORDER BY (service_no ASC, time
ASC);
```

```
cqlsh:tranz> CREATE TABLE IF NOT EXISTS tranz.time_table (
         ...   line_name TEXT,
         ...   service_no INT,
         ...   station_name TEXT,
         ...   longitude DOUBLE,
         ...   latitude DOUBLE,
         ...   time INT,
         ...   distance DOUBLE,
         ...   PRIMARY KEY (line_name, service_no, time)
         ... ) WITH CLUSTERING ORDER BY (service_no ASC, time ASC);
```

--7.Insert the values into the table driver .

```
INSERT INTO tranz.driver
  (driver_name, current_position, mobile, password, skill )
  VALUES ('milan', 'Upper Hutt', 211111, 'mm77', { 'Matangi'
})
  IF NOT EXISTS;

INSERT INTO tranz.driver
  (driver_name, current_position, mobile, password, skill )
  VALUES ('pavle', 'Upper Hutt', 213344, 'pm33', { 'Ganz
Mavag', 'Guliver' })
  IF NOT EXISTS;

INSERT INTO tranz.driver
  (driver_name, current_position, mobile, password, skill )
  VALUES ('pondy', 'Wellington', 216677, 'pondy', { 'Matangi',
'Kiwi Rail' })
  IF NOT EXISTS;

INSERT INTO tranz.driver
  (driver_name, current_position, mobile, password, skill )
```

VALUES ('fred', 'Taita', 210031, 'f5566f', { 'Gulliver', 'Ganz Mavag' })
IF NOT EXISTS;

INSERT INTO tranz.driver
(driver_name, current_position, mobile, password, skill)
VALUES ('jane', 'Waikanae', 213141, 'jjjj', { 'Matangi' })
IF NOT EXISTS;

```
cqlsh:tranz> INSERT INTO tranz.driver
         ...   (driver_name, current_position, mobile, password, skill )
         ...   VALUES ('milan', 'Upper Hutt', 211111, 'mm77', { 'Matangi' });
cqlsh:tranz>
cqlsh:tranz> INSERT INTO tranz.driver
         ...   (driver_name, current_position, mobile, password, skill )
         ...   VALUES ('pavle', 'Upper Hutt', 213344, 'pm33', { 'Ganz Mavag', 'Guliver' });
cqlsh:tranz>
cqlsh:tranz> INSERT INTO tranz.driver
         ...   (driver_name, current_position, mobile, password, skill )
         ...   VALUES ('pondy', 'Wellington', 216677, 'pondy', { 'Matangi', 'Kiwi Rail' });
cqlsh:tranz>
cqlsh:tranz> INSERT INTO tranz.driver
         ...   (driver_name, current_position, mobile, password, skill )
         ...   VALUES ('fred', 'Taita', 210031, 'f5566f', { 'Gulliver', 'Ganz Mavag' });
cqlsh:tranz>
cqlsh:tranz> INSERT INTO tranz.driver
         ...   (driver_name, current_position, mobile, password, skill )
         ...   VALUES ('jane', 'Waikanae', 213141, 'jjjj', { 'Matangi' });
```

-- 8 Drivers can change their password. They provide `old_password` and `new_password`. Update the driver's row with `new_password` only if the `old_password` equal with the stored `password`. If the conditions apply, `password` will be equal with `new_password`.

UPDATE tranz.driver
SET password = 'dhy@@EE3#'
WHERE driver_name = 'pondy'
IF password = 'pondy';

```
cqlsh:tranz> UPDATE tranz.driver SET password = 'dhy@@EE3#' WHERE driver_name = 'pondy';
```

-- 9. Drivers can update their `current_position`: (with city name string) `'Wellington'` OR (with vehicle) `vehicle_id` OR (with not available string constant) `'not_available'`. The update process managed by the app, based on the driver's skill and the location of the train.

```
UPDATE tranz.driver
  SET current_position = 'Petone'
  WHERE driver_name = 'pavle';
```

```
cqlsh:tranz> UPDATE tranz.driver
         ...   SET current_position = 'Petone'
         ...   WHERE driver_name = 'pavle';
```

--10. Seed the initial vehicles data.

```
INSERT INTO tranz.vehicle
  (vehicle_id, status, type )          VALUES ('FA1122', 'Upper Hutt', 'Matangi')
  IF NOT EXISTS;

INSERT INTO tranz.vehicle          (vehicle_id, status, type )
  VALUES ('FP8899', 'maintenance', 'Ganz Mavag')       IF NOT EXISTS;

INSERT INTO tranz.vehicle          (vehicle_id, status, type )
  VALUES ('FA4864', 'Wellington', 'Matangi')         IF NOT EXISTS;

INSERT INTO tranz.vehicle          (vehicle_id, status, type )
  VALUES ('KW3300', 'Wellington', 'KiwiRail')        IF NOT EXISTS;
```

```
cqlsh:tranz> INSERT INTO tranz.vehicle
         ...   (vehicle_id, status, type ) VALUES ('FA1122', 'Upper Hutt', 'Matangi');
cqlsh:tranz>
cqlsh:tranz> INSERT INTO tranz.vehicle (vehicle_id, status, type )
         ...   VALUES ('FP8899', 'maintenance', 'Ganz Mavag');
cqlsh:tranz> INSERT INTO tranz.vehicle (vehicle_id, status, type )
         ...   VALUES ('FA4864', 'Wellington', 'Matangi');
cqlsh:tranz>
cqlsh:tranz> INSERT INTO tranz.vehicle (vehicle_id, status, type )
         ...   VALUES ('KW3300', 'Wellington', 'KiwiRail');
```

- 11. Seed `time_table`.

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (north bound)', 1, 'Wellington', 174.7762, -41.2865, 605, 0) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (north bound)', 1, 'Petone', 174.8851, -41.227, 617, 8.3) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (north bound)', 1, 'Waterloo', 174.9081, -41.2092, 625, 13.3) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (north bound)', 1, 'Taita', 174.9608, -41.1798, 634, 19.0) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (north bound)', 1, 'Silverstream', 175.010276, -41.147283, 642, 26.5) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (north bound)', 1, 'Upper Hutt', 175.0708, -41.1244, 650, 34.3) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (north bound)', 11, 'Wellington', 174.7762, -41.2865, 1935, 0) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (south bound)', 12, 'Upper Hutt', 175.0708, -41.1244, 1900, 0) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (south bound)', 12, 'Silverstream', 175.010276, -41.147283, 1907, 7.8) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Hutt Valley Line (south bound)', 12, 'Taita', 174.9608, -41.1798, 1918, 15.03) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Waikanae Line (north bound)', 5, 'Wellington', 174.7762, -41.2865, 1025, 0) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Waikanae Line (north bound)', 5, 'Paekakariki', 174.951, -40.9881, 1059, 33.1) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Waikanae Line (north bound)', 5, 'Paraparaumu', 175.0084, -40.9142, 1118, 51.3) IF NOT EXISTS;

INSERT INTO tranz.time_table (line_name, service_no, station_name, longitude, latitude, time, distance) VALUES ('Waikanae Line (north bound)', 5, 'Waikanae', 175.0668, -40.8755, 1139, 62.8) IF NOT EXISTS;

-- 12. Read timetable data for showing timetable for passengers. Requested columns from `time_table` table: `line_name`, `station_name`, `time`.

SELECT line_name, station_name, time FROM tranz.time_table;

```
cqlsh:tranz> SELECT line_name, station_name, time FROM tranz.time_table;

 line_name                      | station_name | time
--------------------------------+--------------+------
 Hutt Valley Line (north bound) |   Wellington |  605
 Hutt Valley Line (north bound) |       Petone |  617
 Hutt Valley Line (north bound) |     Waterloo |  625
 Hutt Valley Line (north bound) |        Taita |  634
 Hutt Valley Line (north bound) | Silverstream |  642
 Hutt Valley Line (north bound) |   Upper Hutt |  650
 Hutt Valley Line (north bound) |   Wellington | 1935
 Hutt Valley Line (south bound) |   Upper Hutt | 1900
 Hutt Valley Line (south bound) | Silverstream | 1907
 Hutt Valley Line (south bound) |        Taita | 1918
    Waikanae Line (north bound) |   Wellington | 1025
    Waikanae Line (north bound) |  Paekakariki | 1059
    Waikanae Line (north bound) |  Paraparaumu | 1118
    Waikanae Line (north bound) |     Waikanae | 1139

(14 rows)
```

-- 13. Application can list `station_name`, `service_no`, `time` from `time_table`. `desc` sorted by `time`.
-- The sorting is ASC in this table, the data will be provided as ordered by ASC.

SELECT station_name, service_no, time FROM tranz.time_table;

```
cqlsh:tranz> SELECT station_name, service_no, time FROM tranz.time_table;

 station_name | service_no | time
--------------+------------+------
   Wellington |          1 |  605
        Petone |          1 |  617
     Waterloo |          1 |  625
        Taita |          1 |  634
 Silverstream |          1 |  642
   Upper Hutt |          1 |  650
   Wellington |         11 | 1935
   Upper Hutt |         12 | 1900
 Silverstream |         12 | 1907
        Taita |         12 | 1918
   Wellington |          5 | 1025
  Paekakariki |          5 | 1059
  Paraparaumu |          5 | 1118
     Waikanae |          5 | 1139

(14 rows)
```

-- 14. The iPhone app, which is on the train can read `station_name`, `time`, `line_name`, `service_no..Display the fields when line_name=’ 'Hutt Valley Line (north bound)' AND service_no = 11 limiting to the first document.

SELECT station_name, time, line_name, service_no FROM tranz.time_table WHERE line_name = 'Hutt Valley Line (north bound)' AND service_no = 11 LIMIT 1;

```
cqlsh:tranz> SELECT station_name, time, line_name, service_no FROM tranz.time_table WHERE line_name = 'Hutt Valley Line (north bound)' AND service_no = 11 LIMIT 1;

 station_name | time | line_name                      | service_no
--------------+------+--------------------------------+------------
   Wellington | 1935 | Hutt Valley Line (north bound) |         11

(1 rows)
```

--1 5. The application runs a query to list trains on a station. Find the details when current_position = 'Wellington' and skill is 'Matangi

SELECT * FROM tranz.driver WHERE current_position = 'Wellington' and skill CONTAINS 'Matangi' ALLOW FILTERING;

```
cqlsh:tranz> SELECT * FROM tranz.driver WHERE current_position = 'Wellington' and skill CONTAINS 'Matangi' ALLOW FILTERING;

 driver_name | current_position | mobile | password  | skill
-------------+------------------+--------+-----------+-------------------------
       pondy |       Wellington | 216677 | dhy@@EE3# | {'Kiwi Rail', 'Matangi'}

(1 rows)
```

-- 16. -- Return the `driver_name` if the provided authentication data is right, otherwise no matching data, return an empty table.

SELECT driver_name FROM tranz.driver WHERE driver_name = 'pondy' and password = 'dhy@@EE3#';

```
cqlsh:tranz> SELECT driver_name FROM tranz.driver WHERE driver_name = 'pondy' and password = 'dhy@@EE3#';

 driver_name
-------------
       pondy

(1 rows)
```