Lab - 2	SQL_Assignment_2
IT214 Database Management System, Autumn’2020; Instructor: minal_bhise@daiict, TA: mayank@daiict	

Objectives: I) Run complex queries.

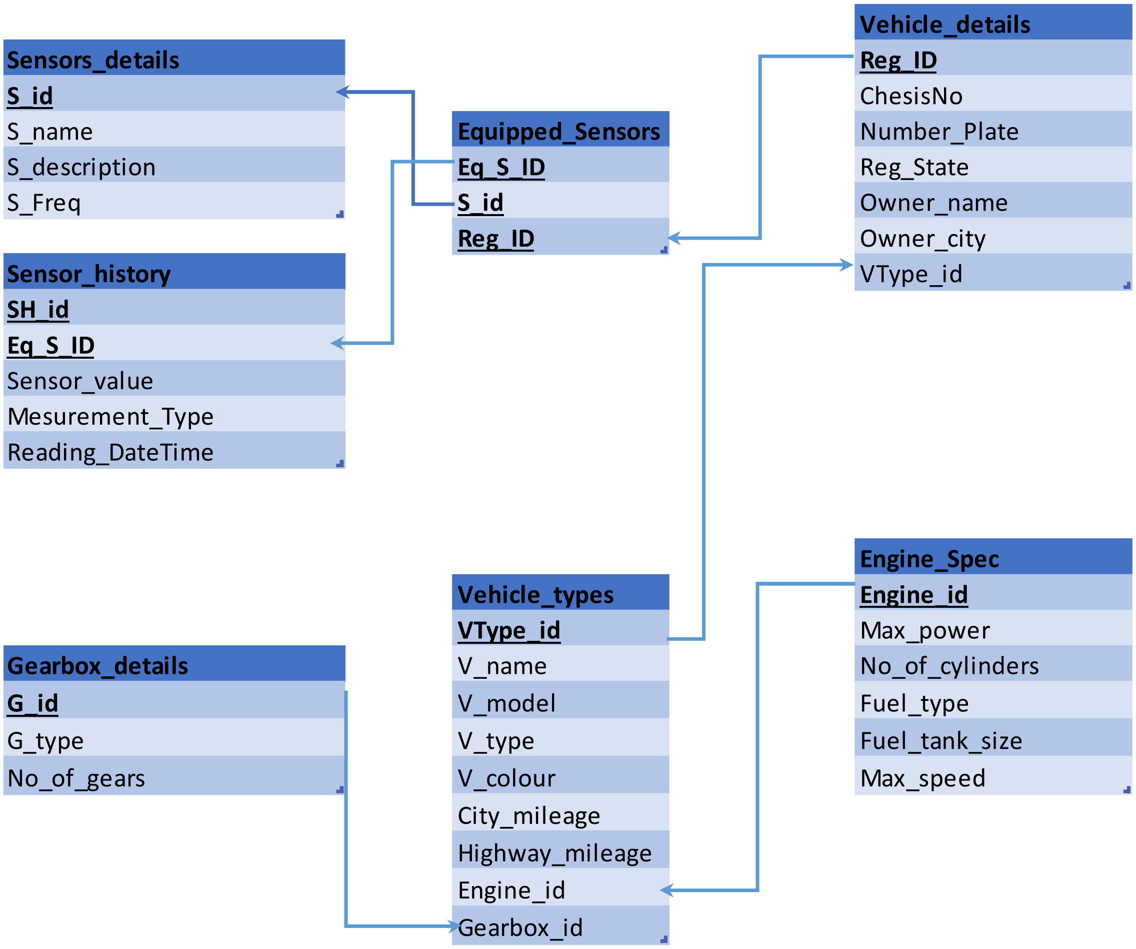
II) Using & Updating VIEWS.

Submission: Each student team needs to upload a **single .pdf** file, which will contain the following things for all the queries listed in your specific section’s lab file.

1) English query and SQL Query in the given sequence.
2) Screenshot of results.
3) Count of tuples in the results.

I. Run complex Queries – Consider Section1’s Schema for the below-given examples.

Section1_Mayank Smart Vehicle Database



Sensors_details: **S_id**, S_name, S_description, S_Freq

Sensor_history: **SH_id**, **Eq_S_ID**, Sensor_value, Mesurement_Type, Reading_DateTime

Equipped_Sensors: **Eq_S_ID**, **S_id**, **Reg_ID**

Vehicle_details: **Reg_ID**, ChesisNo, Number_Plate, Reg_State, Owner_name, Owner_city, VType_id

Gearbox_details: **G_id**, G_type, No_of_gears

Vehicle_types: **VType_id**, V_name, V_model, V_type, V_colour, City_mileage, Highway_mileage, Engine_id, Gearbox_id

Engine_Spec: **Engine_id**, Max_power, No_of_cylinders, Fuel_type, Fuel_tank_size, Max_speed

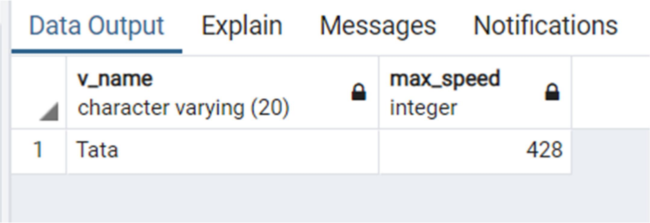
a. **Use of group by.**

```
select count(*), reg_state from vehicle_details group by reg_state;
```

b. Use of JOIN & ORDER BY to find a maximum record with details.

```
SELECT v_name, engine_spec.max_speed
from
sv_db.engine_spec join sv_db.vehicle_types
on
vehicle_types.engine_id = engine_spec.engine_id
order by max_speed desc limit 1;
```

But it shows only the first record, not all the v_names having similar speeds.



	v_name character varying (20)	max_speed integer
1	Tata	428

c. Use of a subquery to find a maximum record with details.

Step1.

Write a subquery. Do not attempt to write a full query at once.

```
SELECT max(max_speed)
        FROM sv_db.engine_spec;
```

Step2.

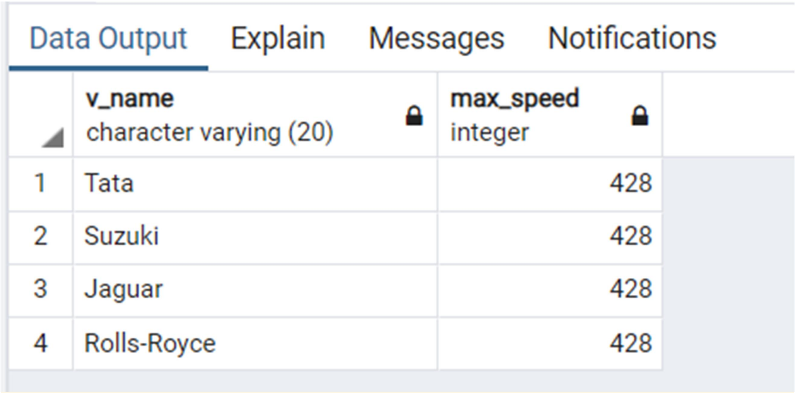
Write a subquery as a part of the main query to get the ID.

```
SELECT engine_id, engine_spec.max_speed
from
sv_db.engine_spec
where
engine_spec.max_speed = (select max(engine_spec.max_speed) from
engine_spec);
```

Step3.

Write a subquery as a part of the main query having joins.

```
SELECT vehicle_types.v_name, engine_spec.max_speed
 from
sv_db.engine_spec join sv_db.vehicle_types
ON
vehicle_types.engine_id = engine_spec.engine_id
where
engine_spec.max_speed = (select max(engine_spec.max_speed) from
engine_spec);
```



	v_name character varying (20)	max_speed integer
1	Tata	428
2	Suzuki	428
3	Jaguar	428
4	Rolls-Royce	428

d. Use of a limit & Order by & SubQuery as Tables to find a maximum record with details.

Step1.

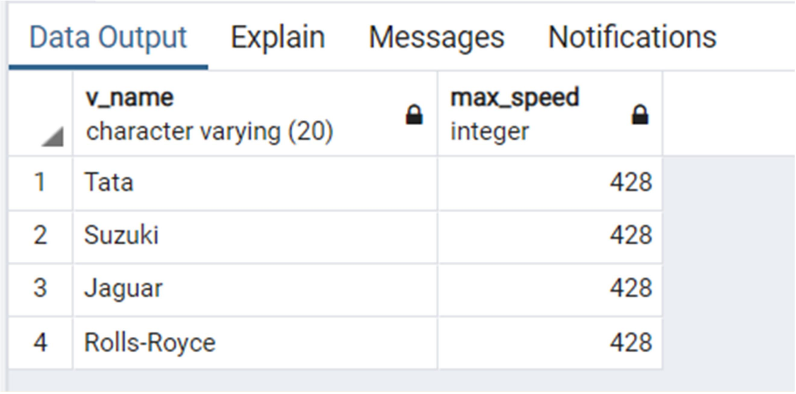
Write a subquery. Do not attempt to write a full query at once.

```
SELECT engine_id, engine_spec.max_speed
from
sv_db.engine_spec
order by max_speed desc limit 1;.
```

Step2.

Join the above query with another table.

```
select vehicle_types.v_name, T1.max_speed from vehicle_types join
(SELECT engine_id, engine_spec.max_speed
from
sv_db.engine_spec
order by max_speed desc limit 1) as T1
on
vehicle_types.engine_id = T1.engine_id;
```



	v_name character varying (20)	max_speed integer
1	Tata	428
2	Suzuki	428
3	Jaguar	428
4	Rolls-Royce	428

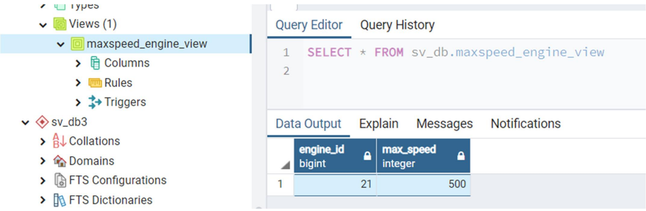
II. Run complex Queries with VIEWs.

a. Use of a View as Tables to find a maximum record with details.

Step1.

First, create a **view**. Once executed

```
CREATE OR REPLACE VIEW MaxSpeed_Engine_VIEW as
SELECT engine_id, engine_spec.max_speed
from
sv_db.engine_spec
order by max_speed desc limit 1;
```



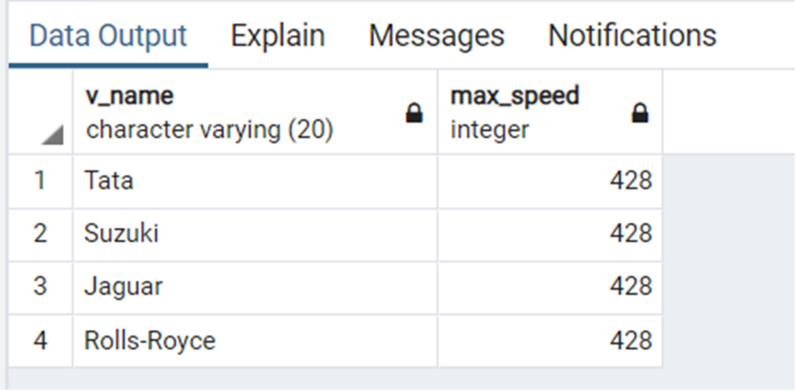
```
SELECT * FROM sv_db.maxspeed_engine_view
```

	engine_id bigint	max_speed integer
1	21	500

Step2.

Use VIEW as a table.

```
select vehicle_types.v_name, max_speed from
vehicle_types
join
MaxSpeed_Engine_VIEW
on
vehicle_types.engine_id = engine_id;
```



	v_name character varying (20)	max_speed integer
1	Tata	428
2	Suzuki	428
3	Jaguar	428
4	Rolls-Royce	428

b. Auto-updates of VIEWs.

```
CREATE OR REPLACE VIEW AVG_Speed_VIEW as
Select AVG(max_speed) from
sv_db.engine_spec;
```

After creating a view, update the max_speed column by adding new records or updating existing records, and see if the average gets updated or not.

```
INSERT INTO sv_db.view_engine_spec(
engine_id, max_power, no_of_cylinders, fuel_type, fuel_tank_size, max_speed)
        VALUES
(22, 222, 22, 'Petrol', 30, 500);
```

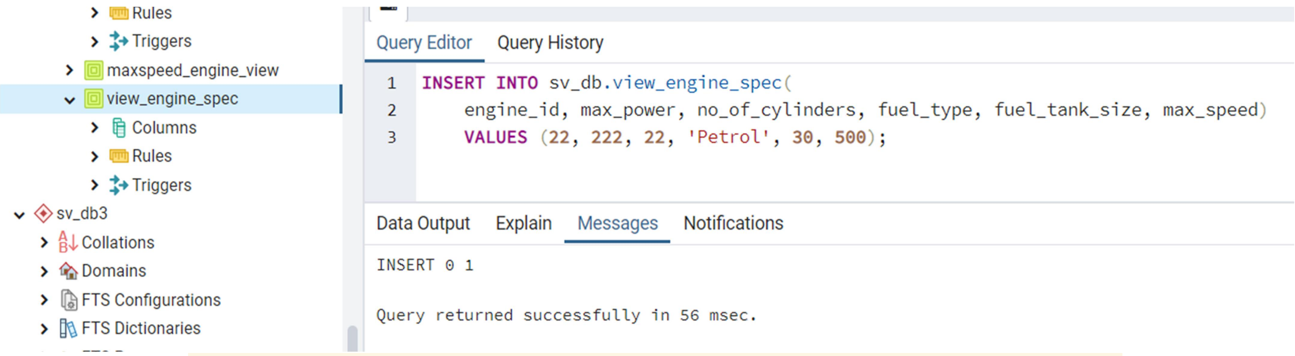
The view will automatically update the data.

```
Select * from AVG_Speed_VIEW;
```

c. Updating VIEW data manually.

```
CREATE OR REPLACE VIEW VIEW_ENGINE_SPEC as
Select * from
sv_db.engine_spec;
```

```
INSERT INTO sv_db.view_engine_spec(
engine_id, max_power, no_of_cylinders, fuel_type, fuel_tank_size, max_speed)
        VALUES
(25, 222, 22, 'Petrol', 30, 550);
```



```
INSERT INTO sv_db.view_engine_spec(
    engine_id, max_power, no_of_cylinders, fuel_type, fuel_tank_size, max_speed)
    VALUES (22, 222, 22, 'Petrol', 30, 500);
```

```
INSERT 0 1

Query returned successfully in 56 msec.
```

Check if Engine_spec has new record or not. As it's a simple view, it will update the original table.

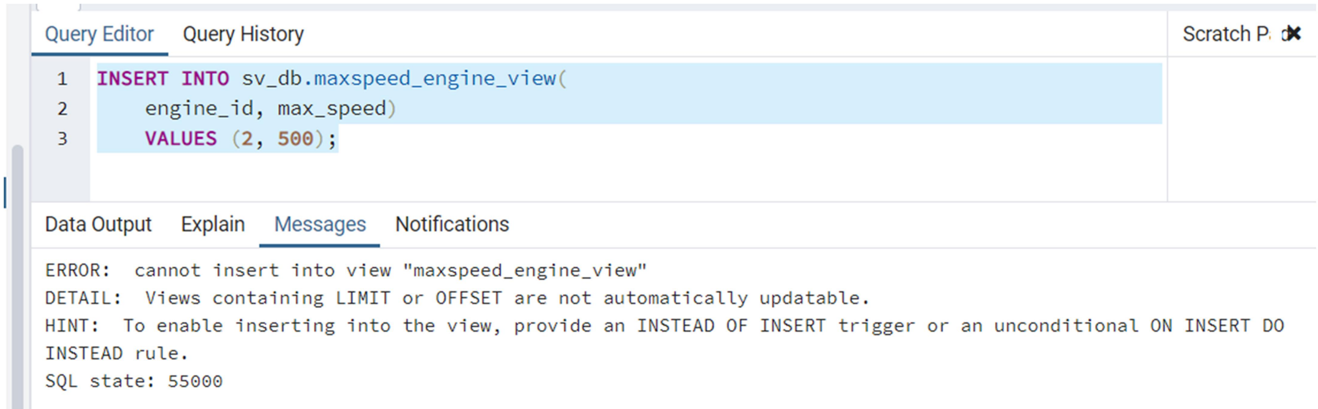
```
SELECT * FROM sv_db.engine_spec
ORDER BY engine_id ASC
```

d. Updating VIEW data manually.

```
CREATE OR REPLACE VIEW maxspeed_engine_view as
Select engine_id, max_speed from
sv_db.engine_spec limit 1;
```

```
INSERT INTO sv_db.maxspeed_engine_view(
        engine_id, max_speed)
        VALUES (2, 500);
```

As it's not a simple view, it will not update the original table.



```
INSERT INTO sv_db.maxspeed_engine_view(
    engine_id, max_speed)
    VALUES (2, 500);
```

```
ERROR:  cannot insert into view "maxspeed_engine_view"
DETAIL:  Views containing LIMIT or OFFSET are not automatically updatable.
HINT:  To enable inserting into the view, provide an INSTEAD OF INSERT trigger or an unconditional ON INSERT DO
INSTEAD rule.
SQL state: 55000
```

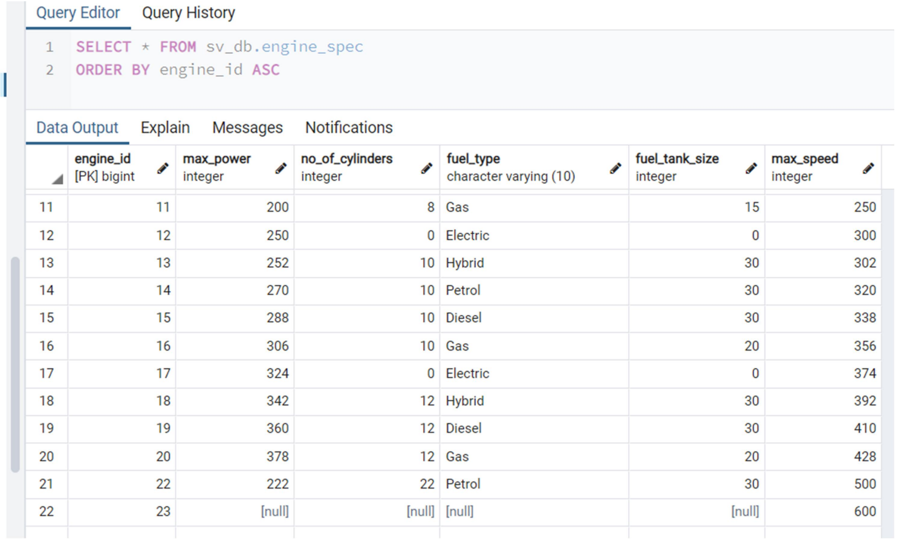
e. **Updating Partial VIEW of a Table which has PK & FK attributes data manually.**

**CREATE OR REPLACE VIEW sv_db.VIEW2_ENGINE_SPEC as
Select engine_id, max_speed from
sv_db.engine_spec;**

Try adding new records to check the original table.

```
INSERT INTO sv_db.VIEW2_ENGINE_SPEC (
        engine_id, max_speed)
        VALUES (2, 500);
```

It will add NULL values in other columns.



```
SELECT * FROM sv_db.engine_spec
ORDER BY engine_id ASC
```

	engine_id [PK] bigint	max_power integer	no_of_cylinders integer	fuel_type character varying (10)	fuel_tank_size integer	max_speed integer
11	11	200	8	Gas	15	250
12	12	250	0	Electric	0	300
13	13	252	10	Hybrid	30	302
14	14	270	10	Petrol	30	320
15	15	288	10	Diesel	30	338
16	16	306	10	Gas	20	356
17	17	324	0	Electric	0	374
18	18	342	12	Hybrid	30	392
19	19	360	12	Diesel	30	410
20	20	378	12	Gas	20	428
21	22	222	22	Petrol	30	500
22	23	[null]	[null]	[null]	[null]	600

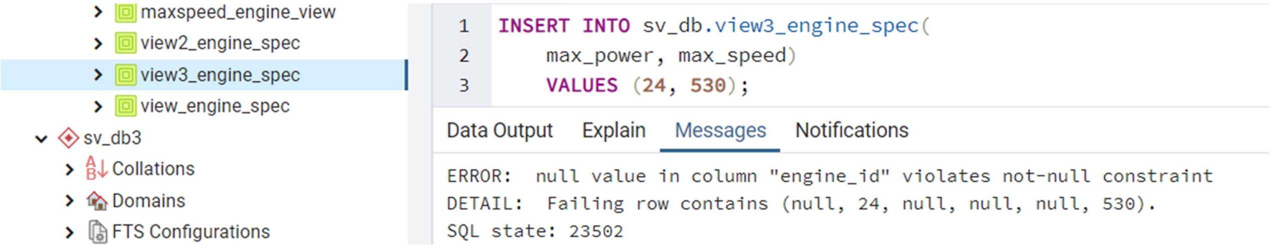
f. **Create & Update below given Partial VIEW of a Table which donot include PK attributes.**

**CREATE OR REPLACE VIEW sv_db.VIEW3_ENGINE_SPEC as
Select max_power, max_speed from sv_db.engine_spec;**

Try adding new records to check the original table.

```
INSERT INTO sv_db.view3_engine_spec(
        max_power, max_speed)
        VALUES (24, 530);
```

It will show an error.



```
INSERT INTO sv_db.view3_engine_spec(
    max_power, max_speed)
    VALUES (24, 530);
```

```
ERROR:  null value in column "engine_id" violates not-null constraint
DETAIL:  Failing row contains (null, 24, null, null, null, 530).
SQL state: 23502
```