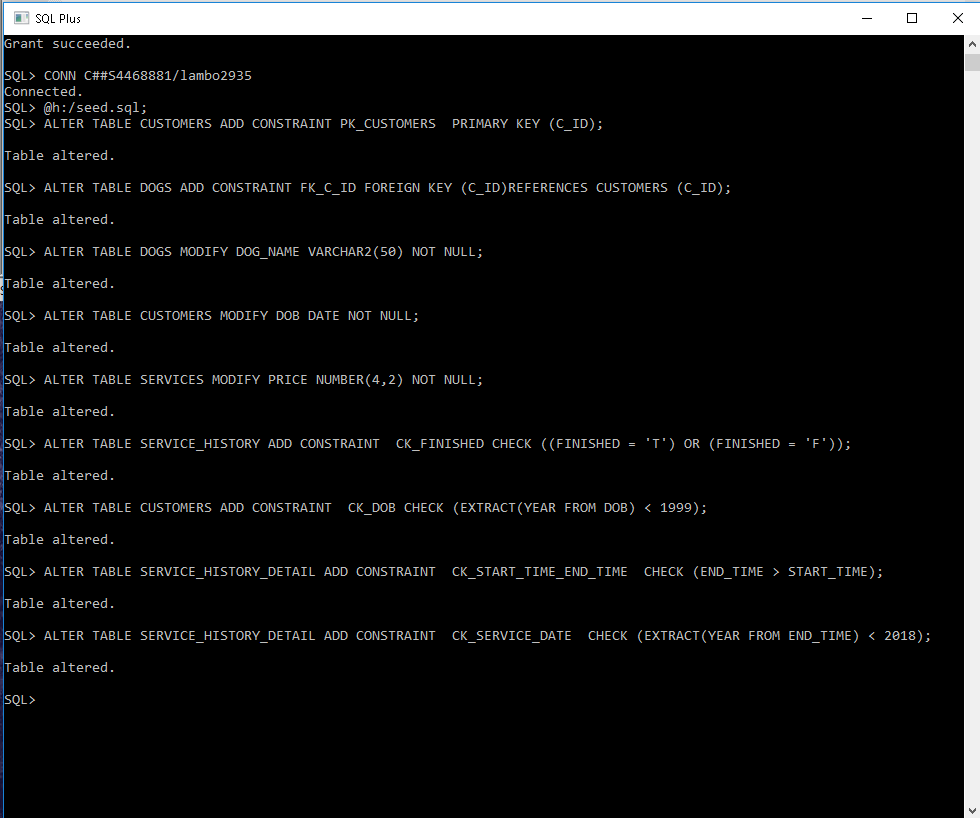
# Task 1: Database Constraints

## A: All Constraints on database

## B: Missing Constraints

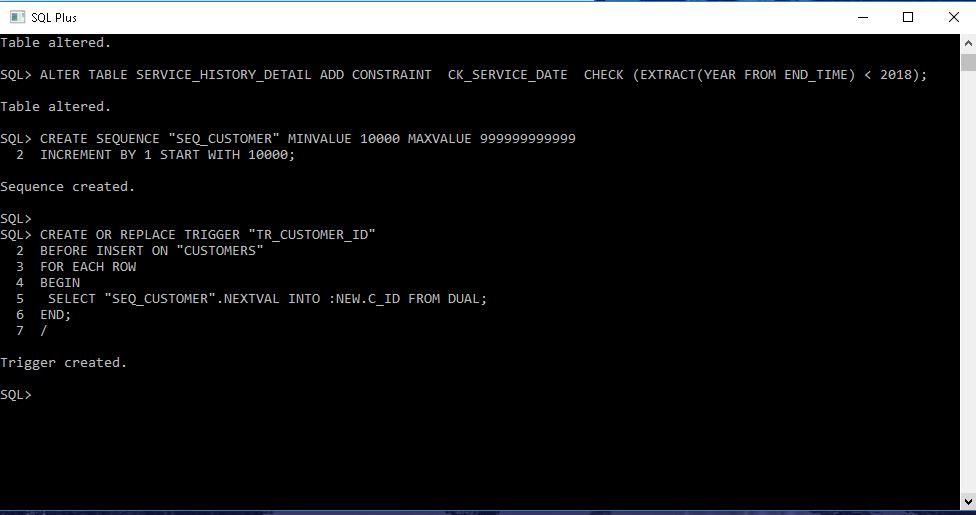
**ALTER TABLE** CUSTOMERS   
**ADD CONSTRAINT** PK\_CUSTOMERS **PRIMARY KEY** (C\_ID);   
  
**ALTER TABLE** DOGS **ADD CONSTRAINT** FK\_C\_ID **FOREIGN KEY** (C\_ID)  
**REFERENCES** CUSTOMERS (C\_ID);   
  
**ALTER TABLE** DOGS **MODIFY** DOG\_NAME VARCHAR2(50) **NOT NULL**;  
  
**ALTER TABLE** CUSTOMERS **MODIFY** DOB **DATE NOT NULL**;  
  
**ALTER TABLE** SERVICES **MODIFY** PRICE **NUMBER**(4,2) **NOT NULL**;  
  
**ALTER TABLE** SERVICE\_HISTORY **ADD CONSTRAINT** CK\_FINISHED   
**CHECK** ((FINISHED = 'T') **OR** (FINISHED = 'F'));   
  
**ALTER TABLE** CUSTOMERS **ADD CONSTRAINT** CK\_DOB   
**CHECK** (*EXTRACT*(**YEAR FROM** DOB) < 1999);   
  
**ALTER TABLE** SERVICE\_HISTORY\_DETAIL **ADD CONSTRAINT** CK\_START\_TIME\_END\_TIME   
**CHECK** (END\_TIME > START\_TIME);   
  
**ALTER TABLE** SERVICE\_HISTORY\_DETAIL **ADD CONSTRAINT** CK\_SERVICE\_DATE   
**CHECK** (*EXTRACT*(**YEAR FROM** END\_TIME) < 2018);



# Task 2: Triggers

## A: SEQ\_CUSTOMER AND TR\_CUSTOMER\_ID

**CREATE** SEQUENCE "SEQ\_CUSTOMER" MINVALUE 10000 **MAXVALUE** 999999999999  
INCREMENT **BY** 1 **START WITH** 10000;  
  
**CREATE OR REPLACE TRIGGER** "TR\_CUSTOMER\_ID"  
**BEFORE INSERT ON** "CUSTOMERS"  
**FOR EACH ROW  
BEGIN  
 SELECT** "SEQ\_CUSTOMER".NEXTVAL **INTO** :NEW.C\_ID **FROM DUAL**;  
**END**;  
/



## B: SEQ\_SERVICE\_HISTORY AND TR\_SERVICE\_ID

**CREATE** SEQUENCE "SEQ\_SERVICE\_HISTORY" MINVALUE 125000 **MAXVALUE** 999999999999  
INCREMENT **BY** 1 **START WITH** 125000;  
  
**CREATE OR REPLACE TRIGGER** "TR\_SERVICE\_ID"  
**BEFORE INSERT ON** "SERVICE\_HISTORY"  
**FOR EACH ROW  
BEGIN  
 SELECT** "SEQ\_SERVICE\_HISTORY".NEXTVAL **INTO** :NEW.SERVICE\_ID **FROM DUAL**;  
**END**;  
/

## C: TR\_SERVICE\_HISTORY\_MESSAGE

**CREATE OR REPLACE TRIGGER** "TR\_SERVICE\_HISTORY\_MESSAGE"  
**BEFORE INSERT OR UPDATE ON** SERVICE\_HISTORY  
**FOR EACH ROW  
DECLARE** CUSTOMER\_F VARCHAR2(50);  
 CUSTOMER\_L VARCHAR2(50);  
 DOG VARCHAR2(50);  
 BREED VARCHAR2(50);  
 STORE VARCHAR2(50);  
 STORE\_MESSAGE VARCHAR2(200);  
**BEGIN  
  
 SELECT DISTINCT** F\_NAME   
 **INTO** CUSTOMER\_F   
 **FROM** CUSTOMERS, DOGS, SERVICE\_HISTORY   
 **WHERE** DOGS.DOG\_ID= :NEW.DOG\_ID  
 **AND** DOGS.C\_ID = CUSTOMERS.C\_ID;  
   
 **SELECT DISTINCT** L\_NAME   
 **INTO** CUSTOMER\_L   
 **FROM** CUSTOMERS, DOGS, SERVICE\_HISTORY   
 **WHERE** DOGS.DOG\_ID= :NEW.DOG\_ID  
 **AND** DOGS.C\_ID = CUSTOMERS.C\_ID;  
   
 **SELECT DISTINCT** DOG\_NAME   
 **INTO** DOG   
 **FROM** DOGS, SERVICE\_HISTORY   
 **WHERE** DOGS.DOG\_ID= :NEW.DOG\_ID;  
   
 **SELECT DISTINCT** DOG\_BREED   
 **INTO** BREED   
 **FROM** DOGS, SERVICE\_HISTORY   
 **WHERE** DOGS.DOG\_ID= :NEW.DOG\_ID;  
   
 **SELECT DISTINCT** STORE\_AREA   
 **INTO** STORE   
 **FROM** STORES, SERVICE\_HISTORY   
 **WHERE** STORES.STORE\_ID= :NEW.STORE\_ID;  
  
 **IF** (:NEW.FINISHED ='T') **THEN** STORE\_MESSAGE := ' is ready for pick-up at ' || STORE || '.';  
 **ELSE** STORE\_MESSAGE := ' is not ready to be picked up yet.';  
 **END IF**;  
  
 :NEW.MESSAGE := 'Hi ' || CUSTOMER\_F || ' ' || CUSTOMER\_L || ' your dog ' || DOG || ' of breed: ' || BREED || ' ' || STORE\_MESSAGE;  
  
 **END**;  
/

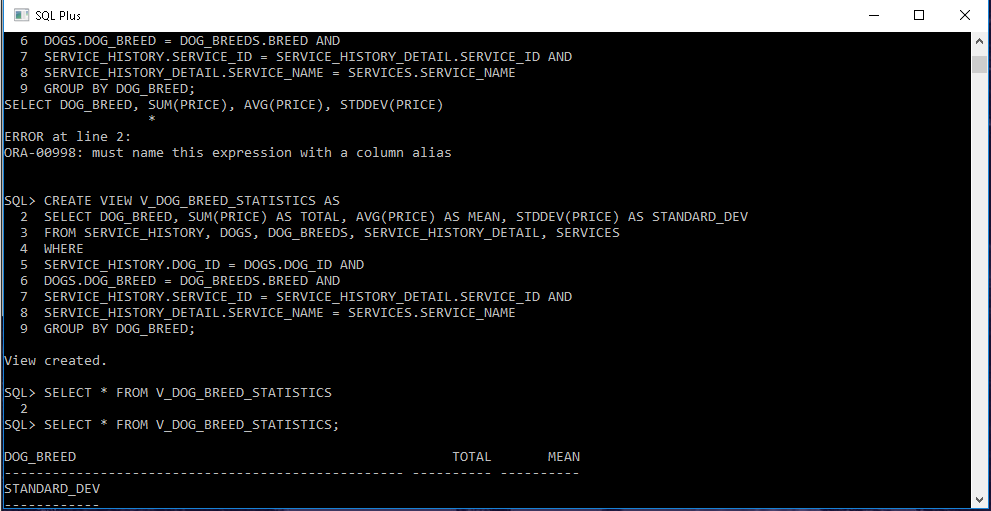
## D: TRIGGER TESTING

**INSERT INTO** CUSTOMERS (F\_NAME, L\_NAME, DOB)  
**VALUES** (‘Luke’, ‘Cheung’, ‘08-OCT-1996’);  
  
**INSERT INTO** SERVICE\_HISTORY (DOG\_ID, STORE\_ID, FINISHED)  
**VALUES** (1234, 30, 'F');

# Task 3: VIEWS

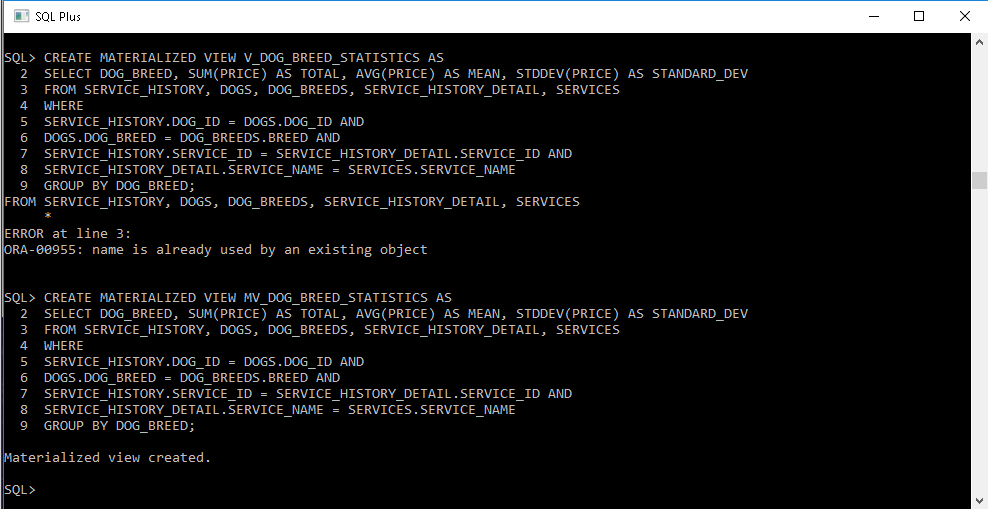
## A: V\_DOG\_BREED\_STATISTICS

**CREATE VIEW** V\_DOG\_BREED\_STATISTICS **AS  
SELECT** DOG\_BREED, *SUM*(PRICE) **AS** TOTAL, *AVG*(PRICE) **AS** MEAN, *STDDEV*(PRICE) **AS** STANDARD\_DEV  
**FROM** SERVICE\_HISTORY, DOGS, DOG\_BREEDS, SERVICE\_HISTORY\_DETAIL, SERVICES  
**WHERE**SERVICE\_HISTORY.DOG\_ID = DOGS.DOG\_ID **AND**DOGS.DOG\_BREED = DOG\_BREEDS.BREED **AND**SERVICE\_HISTORY.SERVICE\_ID = SERVICE\_HISTORY\_DETAIL.SERVICE\_ID **AND**SERVICE\_HISTORY\_DETAIL.SERVICE\_NAME = SERVICES.SERVICE\_NAME  
**GROUP BY** DOG\_BREED;

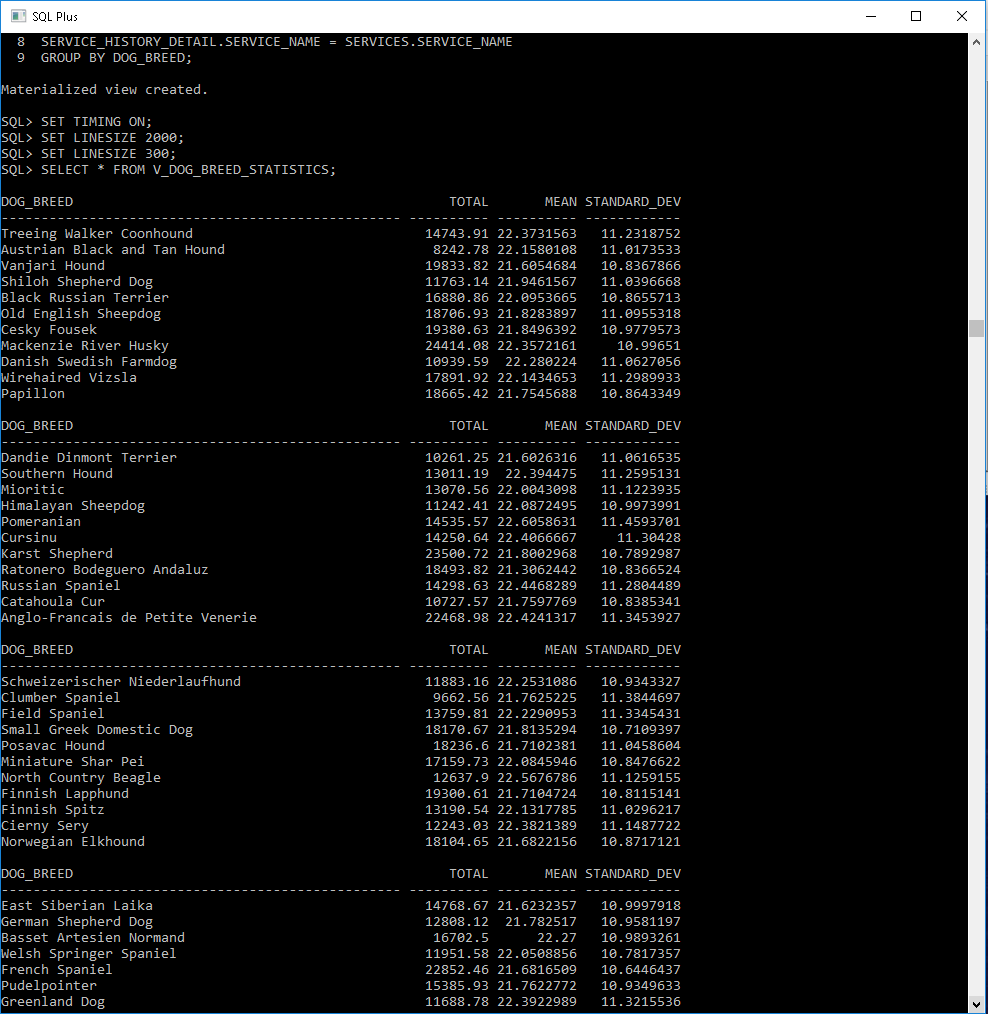


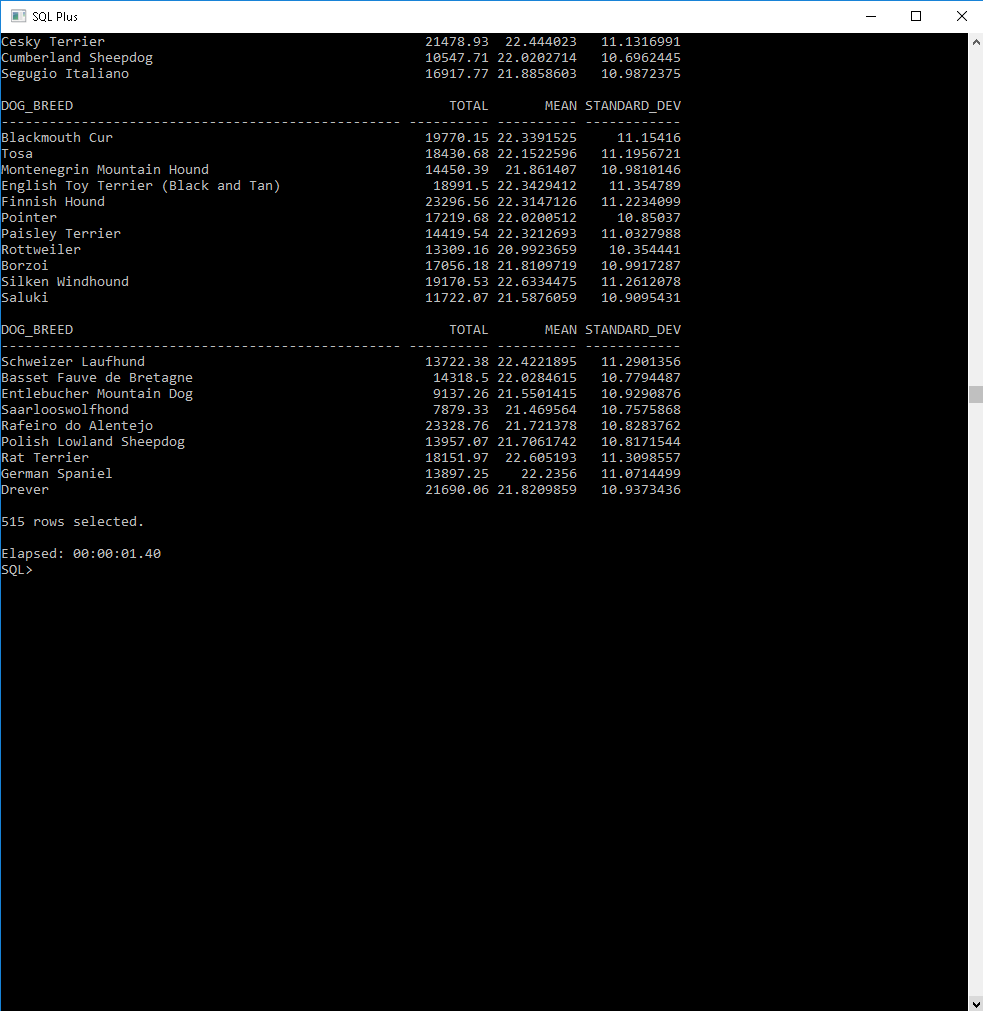
## B: MV\_DOG\_BREED\_STATISTICS

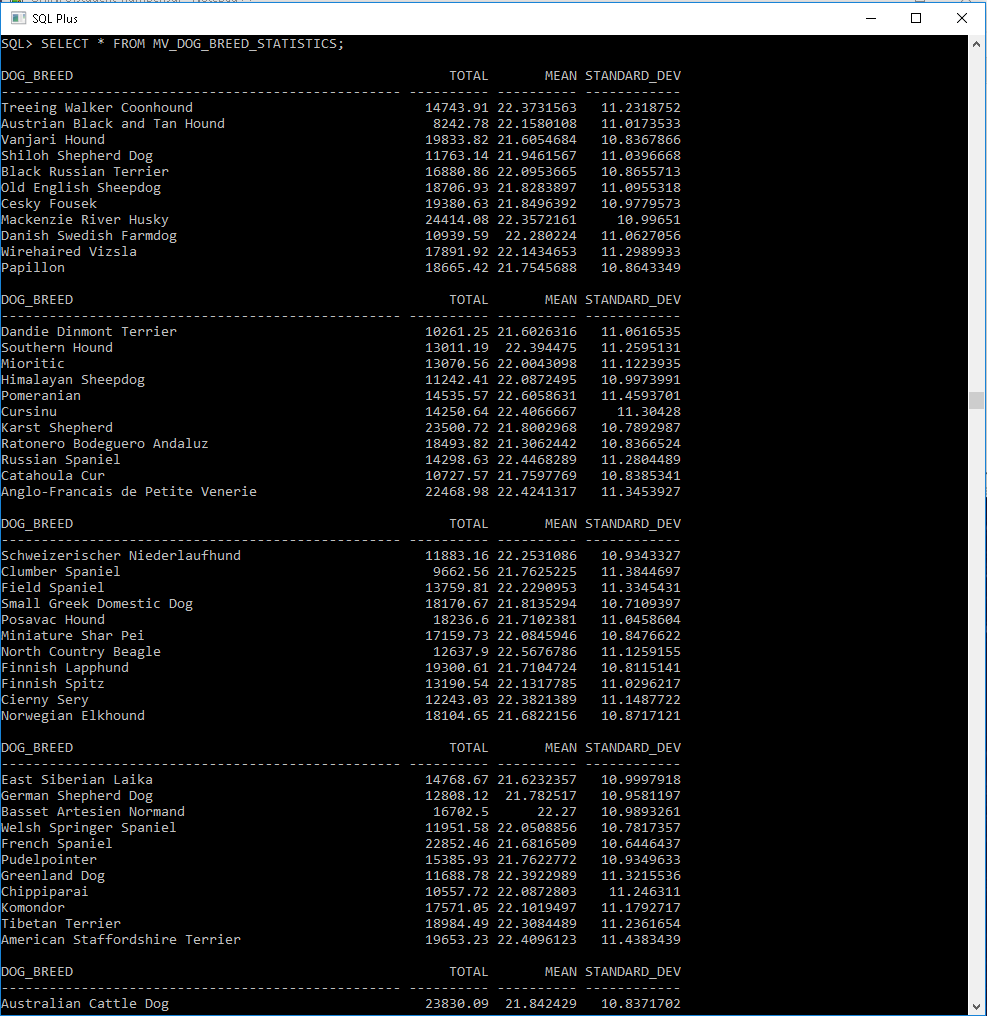
**CREATE** MATERIALIZED **VIEW** MV\_DOG\_BREED\_STATISTICS **AS  
SELECT** DOG\_BREED, *SUM*(PRICE) **AS** TOTAL, *AVG*(PRICE) **AS** MEAN, *STDDEV*(PRICE) **AS** STANDARD\_DEV  
**FROM** SERVICE\_HISTORY, DOGS, DOG\_BREEDS, SERVICE\_HISTORY\_DETAIL, SERVICES  
**WHERE**SERVICE\_HISTORY.DOG\_ID = DOGS.DOG\_ID **AND**DOGS.DOG\_BREED = DOG\_BREEDS.BREED **AND**SERVICE\_HISTORY.SERVICE\_ID = SERVICE\_HISTORY\_DETAIL.SERVICE\_ID **AND**SERVICE\_HISTORY\_DETAIL.SERVICE\_NAME = SERVICES.SERVICE\_NAME  
**GROUP BY** DOG\_BREED;

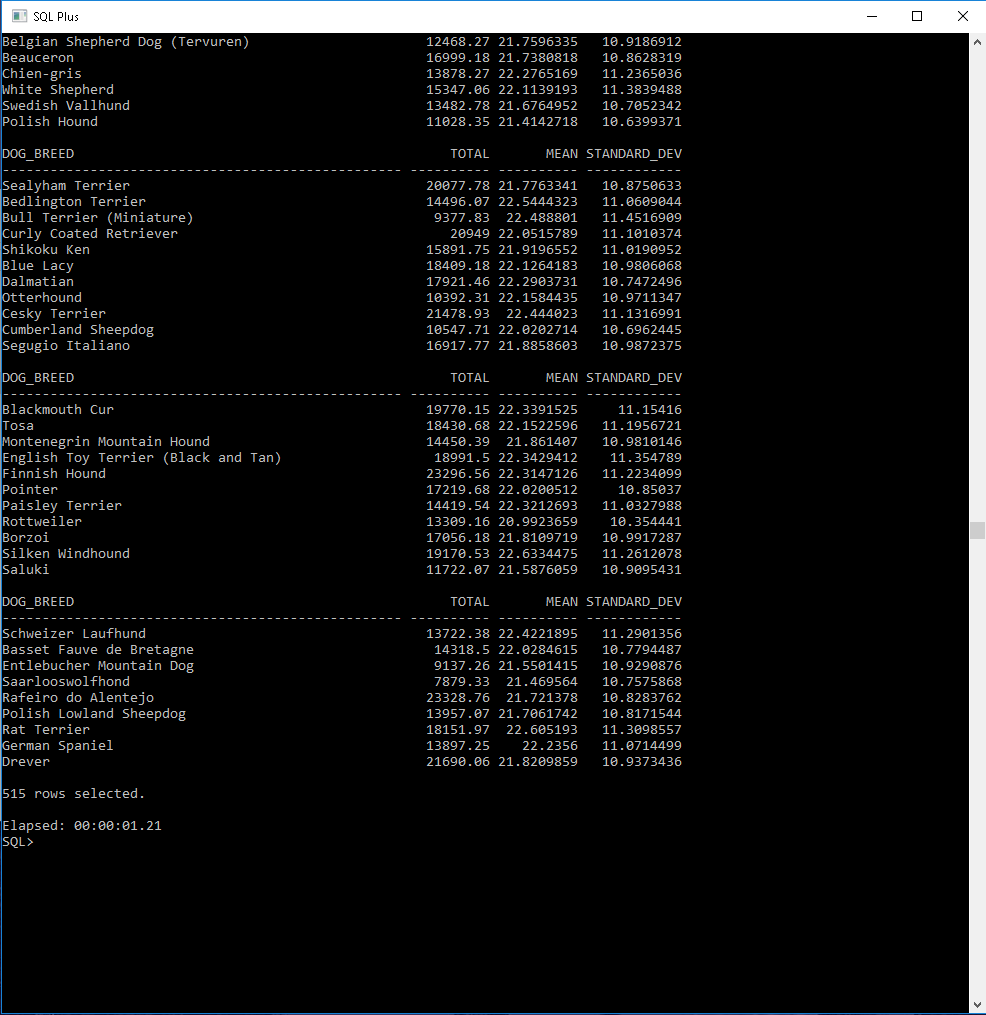


## C: Performance of Materialised views vs Virtual views









The query on Materialised view is running faster than that of Virtual view because Materialised views are separate tables where results of the view queries are saved on physical memory. Whereas, Virtual views are nothing but a save query where the results are displayed by querying on an actual table.

As the size of Materialised view table is much smaller in size than the actual table, the query on Materialized view runs faster as compared to Virtual view where the query is on actual table.

## D: View Updatability

The first view is a virtual view or a “Named Query”. This means that each time a query is made on view, the DBMS gets the result by query modification method and querying on base tables. This ultimately means that this view is always updated as the base table is updated.

On the other hand, the second view is Materialised view which is an actual table with view query results stored on physical memory. This view is not automatically updated whenever there is a change in base tables. In order to update this materialised view, we have to define a refresh policy to trigger when we want the view to get refreshed.

A simple example can be to use “REFRESH FAST ON COMMIT;” command when creating the materialised view to get the view refreshed.

# TASK 4: FUNCTION BASED INDEXES

## A: DENTAL CHECKUP QUERY

**SELECT** \* **FROM** (  
  
**SELECT** DOGS.DOG\_ID, DOGS.DOG\_NAME, STORES.STORE\_ID, STORES.STORE\_AREA, START\_TIME-END\_TIME **AS** TIME\_TAKEN  
**FROM** SERVICE\_HISTORY\_DETAIL, SERVICE\_HISTORY, DOGS, STORES  
**WHERE**SERVICE\_HISTORY\_DETAIL.SERVICE\_ID = SERVICE\_HISTORY.SERVICE\_ID **AND**SERVICE\_HISTORY.STORE\_ID = STORES.STORE\_ID **AND**SERVICE\_HISTORY.DOG\_ID = DOGS.DOG\_ID **AND**SERVICE\_NAME = 'Dental Checkup'  
**ORDER BY** TIME\_TAKEN **DESC**)  
  
**WHERE** ROWNUM=1;

