Anthony Carrick
INFS7903 Assignment

Contents

Task 1: Constraints

1a - SQL to show all constraints:

```
SELECT CONSTRAINT_NAME, TABLE_NAME
FROM USER_CONSTRAINTS
WHERE TABLE_NAME in ('ORGANISATIONS', 'SPOTTERS', 'BIRDS',
'SIGHTINGS');
```

Output

```
CONSTRAINT_NAME                TABLE_NAME
------------------------------ ------------------------------
PK_SPOTTER_ID                  SPOTTERS
PK_ORGANISATION_ID             ORGANISATIONS
PK_BIRD_ID                     BIRDS
```

1b – Implement the missing constraints

```
--Create Sightings Primary Key
ALTER TABLE SIGHTINGS
ADD CONSTRAINT PK_SIGHTING_ID PRIMARY KEY (SIGHTING_ID);

--Create spotters FK
ALTER TABLE SPOTTERS
ADD CONSTRAINT FK_ORG_ID_TO_ORG_ID
FOREIGN KEY (organisation_id) REFERENCES organisations
(organisation_id);

--Create sightings FK
ALTER TABLE SIGHTINGS
ADD CONSTRAINT FK_SPOTTER_ID_TO_SPOTTER_ID
FOREIGN KEY (spotter_id) REFERENCES spotters (spotter_id);

--Create bird FK
ALTER TABLE SIGHTINGS
ADD CONSTRAINT FK_BIRD_ID_TO_BIRD_ID
FOREIGN KEY (bird_id) REFERENCES birds (bird_id);

--create org name not null
ALTER TABLE organisations
ADD CONSTRAINT NN_ORGANISATION_NAME
CHECK (organisation_name IS NOT NULL);

--create spotter name not null
ALTER TABLE SPOTTERS
ADD CONSTRAINT NN_SPOTTER_NAME
CHECK (spotter_name IS NOT NULL);
```

```
--create bird name not null
ALTER TABLE birds
ADD CONSTRAINT NN_BIRD_NAME
CHECK (bird_name IS NOT NULL);

--create bird name not null
ALTER TABLE sightings
ADD CONSTRAINT CK_SIGHTING_DATE
CHECK (sighting_date < TO_DATE('2017-01-01', 'YYYY-MM-DD'));
```

Result (of rerunning the sql in 1a):

```
CONSTRAINT_NAME                TABLE_NAME
------------------------------ ------------------------------
PK_BIRD_ID                     BIRDS
NN_BIRD_NAME                   BIRDS
PK_ORGANISATION_ID             ORGANISATIONS
NN_ORGANISATION_NAME           ORGANISATIONS
CK_SIGHTING_DATE               SIGHTINGS
PK_SIGHTING_ID                 SIGHTINGS
FK_SPOTTER_ID_TO_SPOTTER_ID    SIGHTINGS
FK_BIRD_ID_TO_BIRD_ID          SIGHTINGS
PK_SPOTTER_ID                  SPOTTERS
FK_ORG_ID_TO_ORG_ID            SPOTTERS
NN_SPOTTER_NAME                SPOTTERS

11 rows selected.
```

Task 2: Triggers

2a – create sequence and trigger

--create sightings sequence

```
CREATE SEQUENCE "SEQ_SIGHTINGS"
INCREMENT BY 1 START WITH 300000;

CREATE OR REPLACE TRIGGER "TR_SIGHTING_ID"
      BEFORE INSERT ON "SIGHTINGS"
      FOR EACH ROW
BEGIN
      SELECT "SEQ_SIGHTINGS".NEXTVAL INTO :NEW.SIGHTING_ID FROM
DUAL;
END;
/
```

2b – create description trigger

```
--create sighting sentence formatting trigger
CREATE OR REPLACE TRIGGER "TR_SIGHTING_DESC"
      BEFORE INSERT ON "SIGHTINGS"
      FOR EACH ROW
DECLARE
      BIRDNAME varchar(50);
      X varchar(10);
      Y varchar(10);
BEGIN
      SELECT BIRD_NAME
      INTO BIRDNAME
      FROM BIRDS
      WHERE BIRDS.BIRD_ID = :NEW.bird_id;


      --NORTH OR SOUTH
      IF (:NEW.LATITUDE >= -28.1) THEN
            Y := 'North';
      ELSE
            Y := 'South';
      END IF;
      --east OR west
      IF (:NEW.LONGITUDE <= 151.25) THEN
            X := 'Western';
      ELSE
            X := 'Eastern';
      END IF;

      :NEW.DESCRIPTION := 'A bird of the species ' || BIRDNAME || '
was spotted in the the ' || Y || '-' || X || ' part of the
observation area.';
END;
/
```

2c - Insert

```
INSERT INTO sightings (spotter_id, bird_id, latitude, longitude, sighting_date)
VALUES (2457, 901, -28.0, 152, '09-MAR-2016');


Elapsed: 00:00:00.03
INSERT INTO sightings (spotter_id, bird_id, latitude, longitude, sighting_date)
VALUES (1024, 512, -25.6, 153, '09-MAR-2016');

1 row created.
```

2d - Select All Sightings from 9th March 2016

```
SELECT * FROM SIGHTINGS
WHERE SIGHTING_DATE = '09-MAR-2016';


SIGHTING_ID SPOTTER_ID    BIRD_ID   LATITUDE  LONGITUDE SIGHTING_
----------- ---------- ---------- ---------- ---------- ---------
DESCRIPTION
--------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------
     300000       2457        901        -28        152 09/MAR/16
A bird of the species Australian pied cormorant was spotted in the the North-Eastern part of the
observation area.

     300001       1024        512      -25.6        153 09/MAR/16
A bird of the species Mrs. Humes pheasant was spotted in the the North-Eastern part of the observation
area.
```

Task 3: Views

3a - Virtual View

```
--create view of total bird sightings grouped by organisation
CREATE VIEW V_ORGANISATION_BIRD_COUNT AS
      SELECT organisation_name, count(*) as bird_count
      FROM sightings s, organisations o, spotters sp
      WHERE s.spotter_id = sp.spotter_id and sp.organisation_id = o.organisation_id
      GROUP BY organisation_name;
/
```

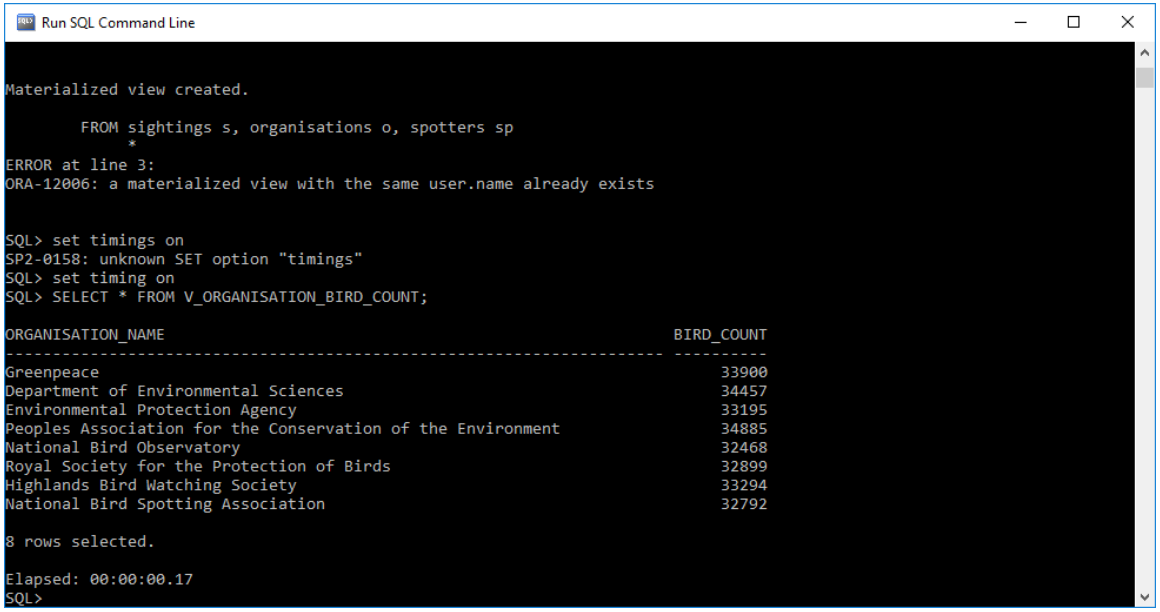
3b - Materialised View

```
--create materialised view of same data as above
CREATE MATERIALIZED VIEW MV_ORGANISATION_BIRD_COUNT AS
      SELECT organisation_name, count(*) as bird_count
      FROM sightings s, organisations o, spotters sp
      WHERE s.spotter_id = sp.spotter_id and sp.organisation_id = o.organisation_id
      GROUP BY organisation_name;
/
```

3c – Run, Compare, Explain

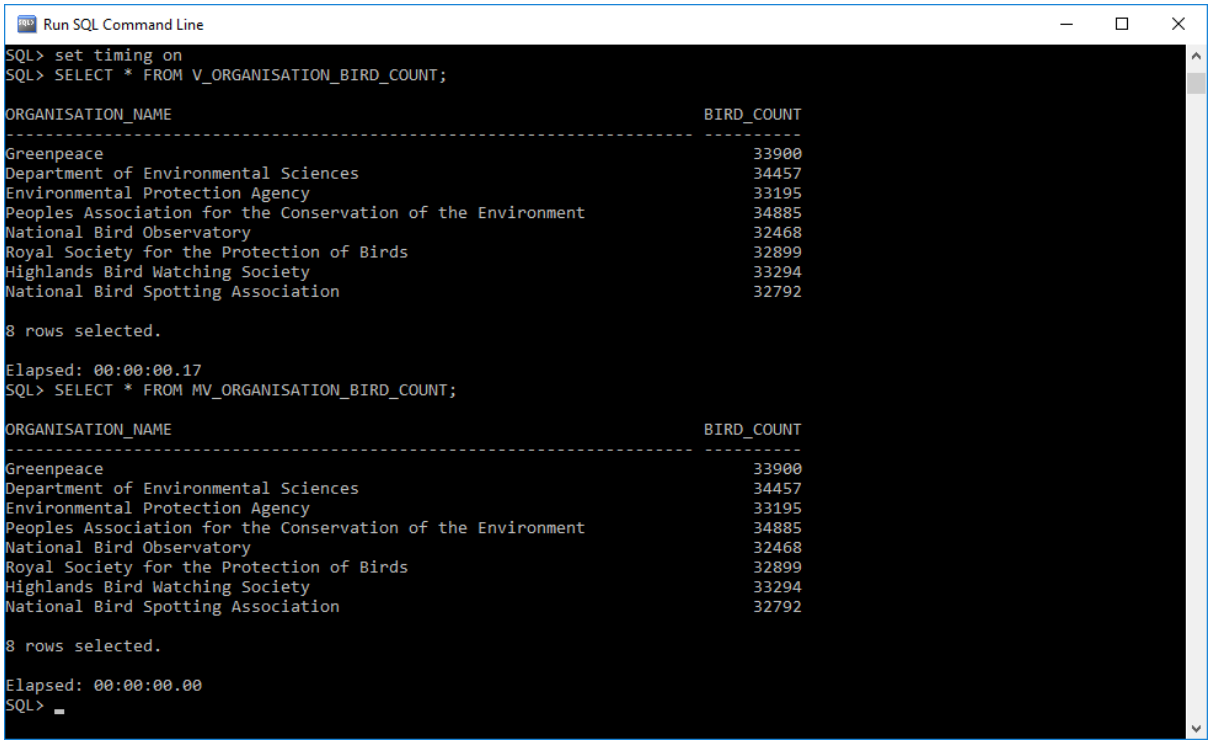
```
SQL> set timing on
SQL> SELECT * FROM V_ORGANISATION_BIRD_COUNT;

ORGANISATION_NAME                                                  BIRD_COUNT
------------------------------------------------------------------ ----------
Greenpeace                                                              33900
Department of Environmental Sciences                                    34457
Environmental Protection Agency                                         33195
Peoples Association for the Conservation of the Environment             34885
```

```
National Bird Observatory                                          32468
Royal Society for the Protection of Birds                          32899
Highlands Bird Watching Society                                    33294
National Bird Spotting Association                                 32792

8 rows selected.

Elapsed: 00:00:00.17
```




```
SQL> SELECT * FROM MV_ORGANISATION_BIRD_COUNT;

ORGANISATION_NAME                                                  BIRD_COUNT
------------------------------------------------------------------ ----------
Greenpeace                                                              33900
Department of Environmental Sciences                                    34457
Environmental Protection Agency                                         33195
Peoples Association for the Conservation of the Environment             34885
```

```
National Bird Observatory                                    32468
Royal Society for the Protection of Birds                    32899
Highlands Bird Watching Society                              33294
National Bird Spotting Association                           32792

8 rows selected.

Elapsed: 00:00:00.00
```



The materialised view's result table is saved to the database catalogue so the DBMS doesn't need to perform the actual search (with joins and reads from several tables), instead it can just read the static data from the disk, likely even compressed and saved in other optimised ways (because it's just in the catalogue).

Querying the original view requires re-running the query each time.

Task 4: Function Based Indexes

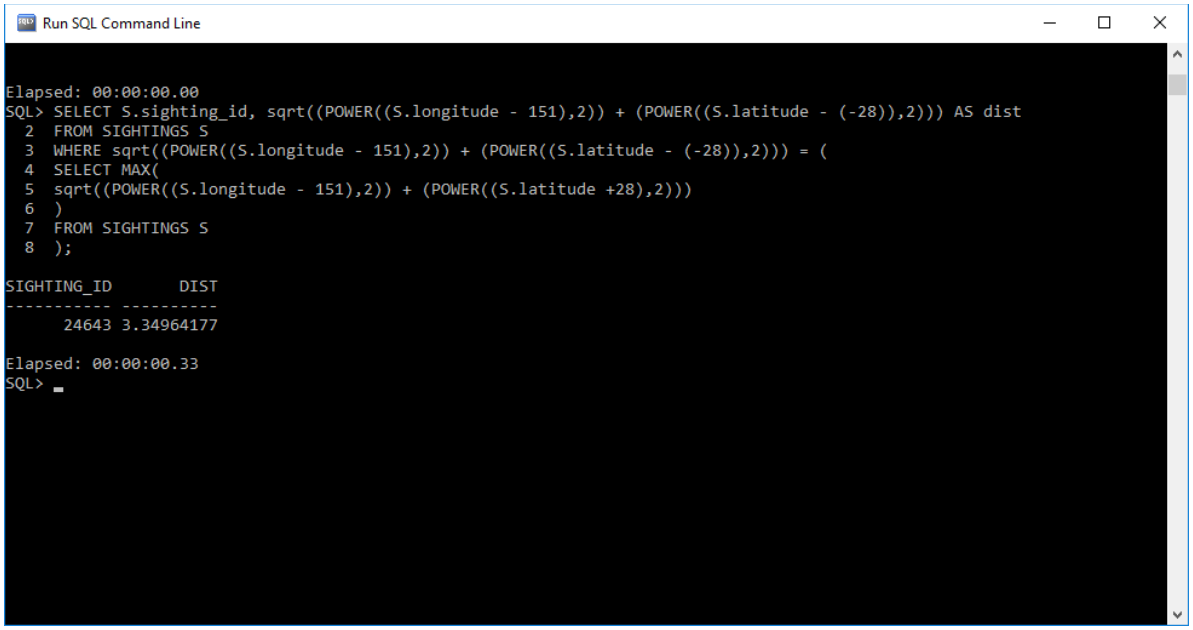
4a - SQL for Sighting with max distance

```
SELECT S.sighting_id, sqrt((POWER((S.longitude - 151),2)) + (POWER((S.latitude - (-28)),2))) AS dist
FROM SIGHTINGS S
WHERE sqrt((POWER((S.longitude - 151),2)) + (POWER((S.latitude - (-28)),2))) = (
      SELECT MAX( sqrt((POWER((S.longitude - 151),2)) + (POWER((S.latitude +28),2))) )
      FROM SIGHTINGS S
      );
```

Result

```
SIGHTING_ID       DIST
----------- ----------
      24643 3.34964177

Elapsed: 00:00:00.33
```

Timings



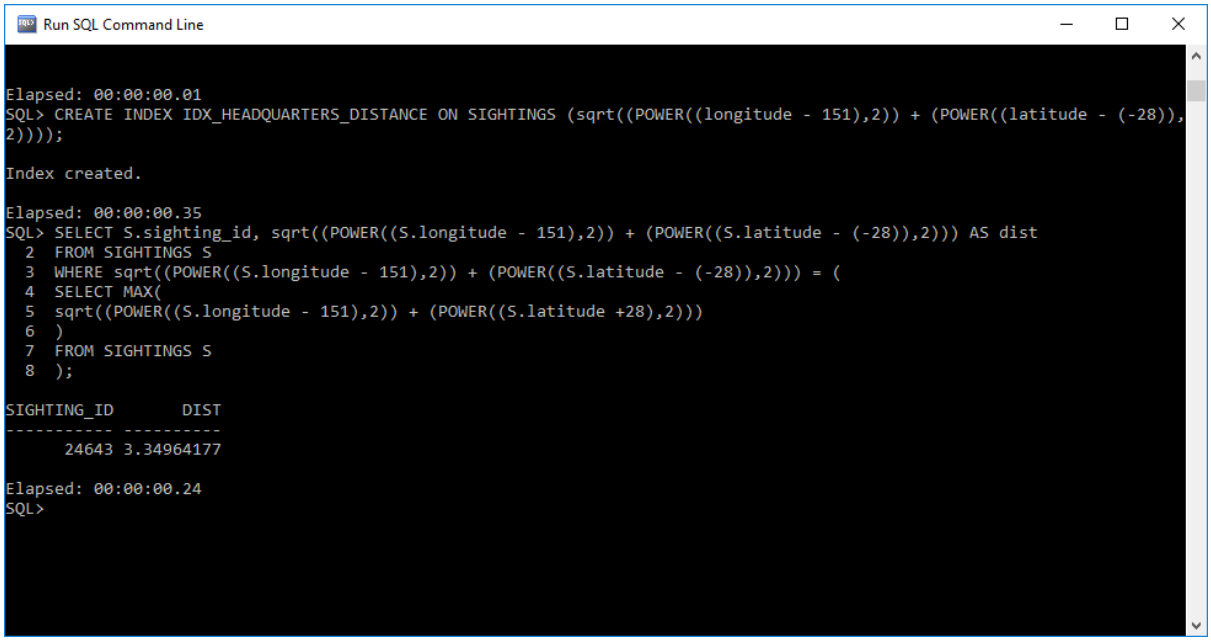
4b - Create index

```
CREATE INDEX IDX_HEADQUARTERS_DISTANCE ON SIGHTINGS (sqrt((POWER((longitude - 151),2)) + (POWER((latitude - (-
28)),2))));

Index created.
```

4c - Rerun Query after making Index

```
SQL> SELECT S.sighting_id, sqrt((POWER((S.longitude - 151),2)) + (POWER((S.latitude - (-28)),2))) AS dist
  2  FROM SIGHTINGS S
  3  WHERE sqrt((POWER((S.longitude - 151),2)) + (POWER((S.latitude - (-28)),2))) = (
  4  SELECT MAX(
  5  sqrt((POWER((S.longitude - 151),2)) + (POWER((S.latitude +28),2)))
  6  )
  7  FROM SIGHTINGS S
  8  );

SIGHTING_ID       DIST
----------- ----------
      24643 3.34964177

Elapsed: 00:00:00.24
```

Timings



Explanation

The index has made the query faster by 90 milliseconds because it has indexed the Distance calculation.

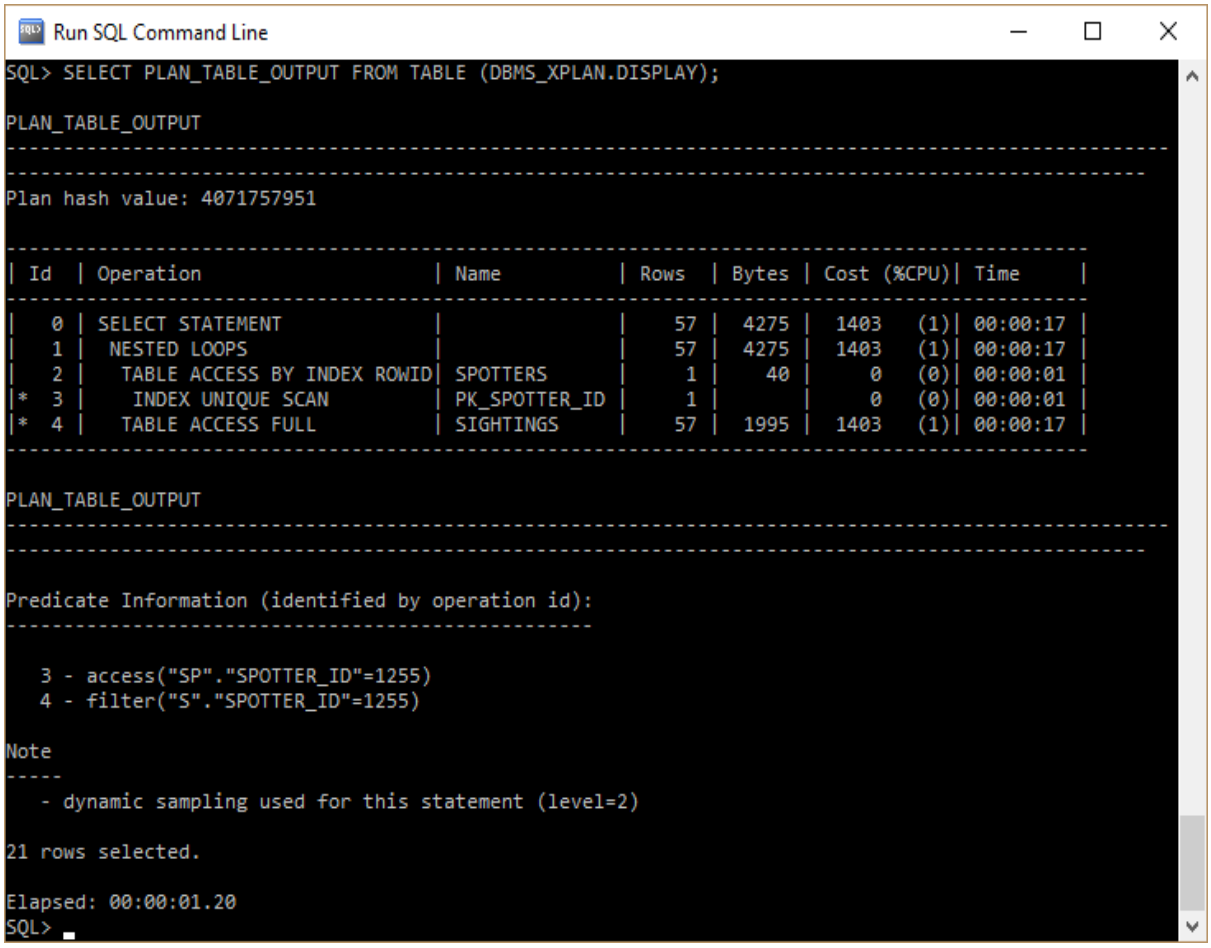
Task 5: Execution Plan & Analysis

5a

```
select sighting_id, spotter_name, sighting_date
from sightings s, spotters sp
where sp.spotter_id = 1255 and s.spotter_id = sp.spotter_id;


SIGHTING_ID SPOTTER_NAME                                       SIGHTING_
----------- -------------------------------------------------- ---------
      21128 Alana Debari                                       07/DEC/06
      25266 Alana Debari                                       29/NOV/01
      42063 Alana Debari                                       06/FEB/03
      47185 Alana Debari                                       20/JAN/04
      44921 Alana Debari                                       28/FEB/03
      59410 Alana Debari                                       22/DEC/04
      78777 Alana Debari                                       25/MAR/11
      88328 Alana Debari                                       11/APR/07

. . . .

     100706 Alana Debari                                       07/APR/04
     106789 Alana Debari                                       28/JUN/00
     247582 Alana Debari                                       25/DEC/10
     247606 Alana Debari                                       21/OCT/14
     248257 Alana Debari                                       20/JUL/15
     249309 Alana Debari                                       04/APR/15
     263323 Alana Debari                                       13/JUN/05

52 rows selected.

Elapsed: 00:00:00.75
```

```
EXPLAIN PLAN FOR
select sighting_id, spotter_name, sighting_date
from sightings s, spotters sp
where sp.spotter_id = 1255 and s.spotter_id = sp.spotter_id;

SELECT PLAN_TABLE_OUTPUT FROM TABLE (DBMS_XPLAN.DISPLAY);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------------------------------------
Plan hash value: 4071757951

-----------------------------------------------------------------------------------------
| Id  | Operation                     | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |               |    57 |  4275 |  1403   (1)| 00:00:17 |
|   1 |  NESTED LOOPS                 |               |    57 |  4275 |  1403   (1)| 00:00:17 |
|   2 |   TABLE ACCESS BY INDEX ROWID| SPOTTERS      |     1 |    40 |     0   (0)| 00:00:01 |
|*  3 |    INDEX UNIQUE SCAN          | PK_SPOTTER_ID |     1 |       |     0   (0)| 00:00:01 |
|*  4 |   TABLE ACCESS FULL           | SIGHTINGS     |    57 |  1995 |  1403   (1)| 00:00:17 |
-----------------------------------------------------------------------------------------

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   3 - access("SP"."SPOTTER_ID"=1255)
   4 - filter("S"."SPOTTER_ID"=1255)

Note
-----
   - dynamic sampling used for this statement (level=2)

21 rows selected.
Elapsed: 00:00:01.20
```

Screenshot



Explanation

- INDEX UNIQUE SCAN - Scan through the index on Spotter ID to find the unique value – occurs first (you can tell since it's the most indented.)
- TABLE ACCESS BY INDEX ROWID using the row ID found in the INDEX UNIQUE SCAN
- SIGHTINGS table is accessed in full in ID 4 as part of the NESTED LOOP in ID 1
- Finally, the SELECT statement returns the listed columns, and rows meeting the criteria in the WHERE clause.

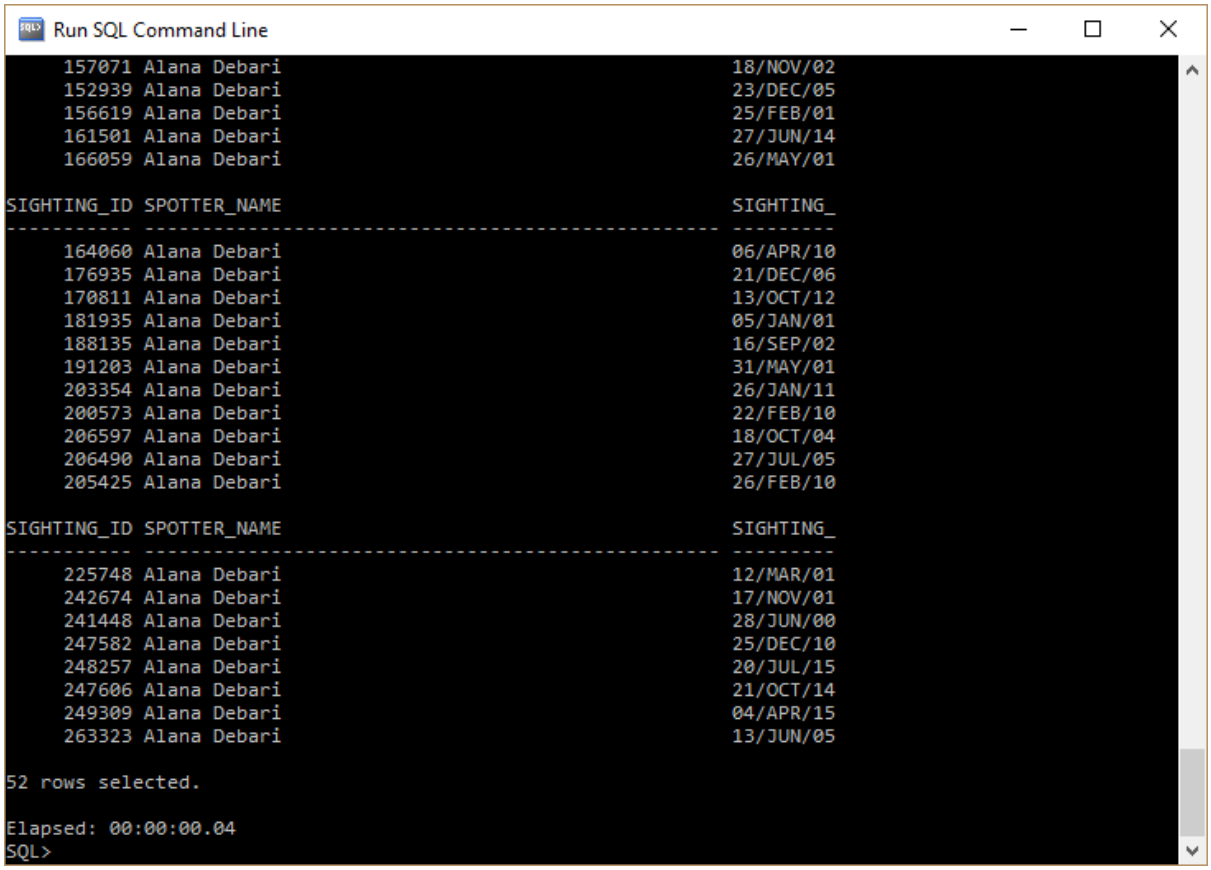
5b – drop the primary key on Spotter ID

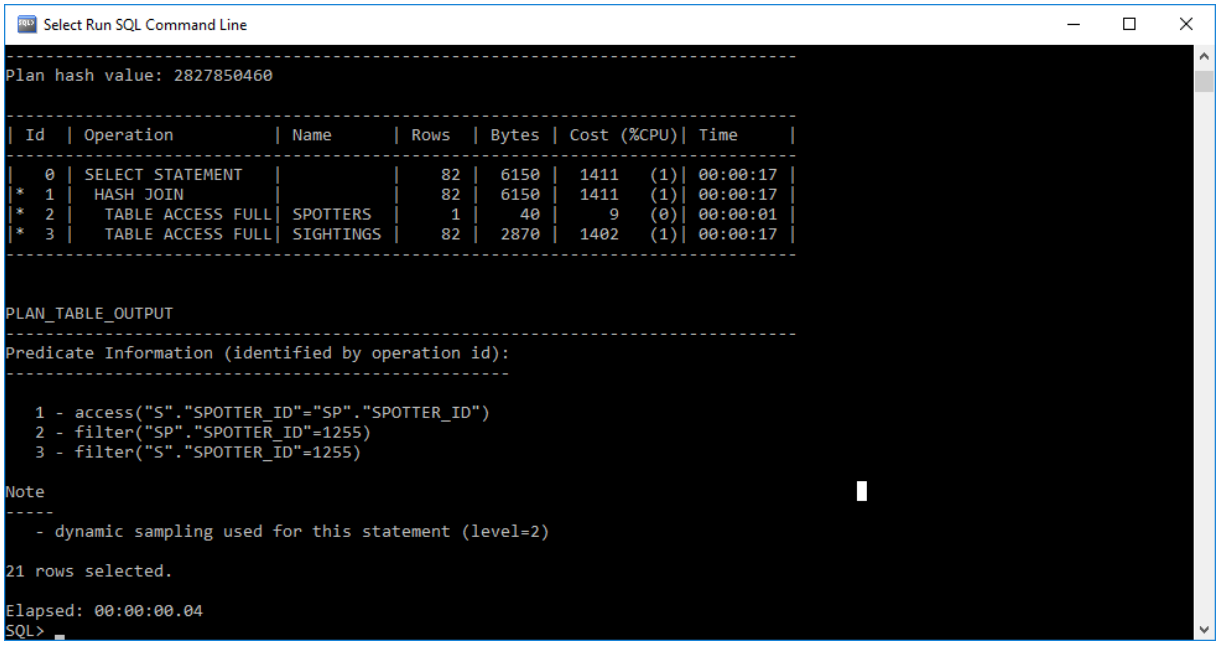
```
ALTER TABLE SPOTTERS
DROP CONSTRAINT PK_SPOTTER_ID;

ERROR at line 2:
ORA-02273: this unique/primary key is referenced by some foreign keys

ALTER TABLE SIGHTINGS
DROP CONSTRAINT FK_SPOTTER_ID_TO_SPOTTER_ID;

Table altered.

ALTER TABLE SPOTTERS
DROP CONSTRAINT PK_SPOTTER_ID;

Table altered.
```

Let's rerun the same query again

```
EXPLAIN PLAN FOR
select sighting_id, spotter_name, sighting_date
from sightings s, spotters sp
where sp.spotter_id = 1255 and s.spotter_id = sp.spotter_id;

SELECT PLAN_TABLE_OUTPUT FROM TABLE (DBMS_XPLAN.DISPLAY);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Plan hash value: 2827850460

--------------------------------------------------------------------------------
| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |           |    82 |  6150 |  1411   (1)| 00:00:17 |
|*  1 |  HASH JOIN         |           |    82 |  6150 |  1411   (1)| 00:00:17 |
|*  2 |   TABLE ACCESS FULL| SPOTTERS  |     1 |    40 |     9   (0)| 00:00:01 |
|*  3 |   TABLE ACCESS FULL| SIGHTINGS |    82 |  2870 |  1402   (1)| 00:00:17 |
--------------------------------------------------------------------------------


PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------

   1 - access("S"."SPOTTER_ID"="SP"."SPOTTER_ID")
   2 - filter("SP"."SPOTTER_ID"=1255)
   3 - filter("S"."SPOTTER_ID"=1255)

Note
-----
   - dynamic sampling used for this statement (level=2)

21 rows selected.
```

```
--Query for spotter_id 1255 again
select sighting_id, spotter_name, sighting_date
from sightings s, spotters sp
where sp.spotter_id = 1255 and s.spotter_id = sp.spotter_id;

SIGHTING_ID SPOTTER_NAME                                       SIGHTING_
----------- -------------------------------------------------- ---------
      21128 Alana Debari                                       07/DEC/06
      25266 Alana Debari                                       29/NOV/01
      42063 Alana Debari                                       06/FEB/03
      47185 Alana Debari                                       20/JAN/04
      44921 Alana Debari                                       28/FEB/03
      59410 Alana Debari                                       22/DEC/04
      78777 Alana Debari                                       25/MAR/11
      88328 Alana Debari                                       11/APR/07

. . . .

     100706 Alana Debari                                       07/APR/04
     106789 Alana Debari                                       28/JUN/00
     247582 Alana Debari                                       25/DEC/10
     247606 Alana Debari                                       21/OCT/14
     248257 Alana Debari                                       20/JUL/15
     249309 Alana Debari                                       04/APR/15
     263323 Alana Debari                                       13/JUN/05

52 rows selected.

Elapsed: 00:00:00.86
```





Explanation and Difference Identification

Query listed for reference:

```
select sighting_id, spotter_name, sighting_date
from sightings s, spotters sp
where sp.spotter_id = 1255 and s.spotter_id = sp.spotter_id;
```

Now that there is no index (indexes are created primary keys automatically) on Spotter ID, the DMBS has to access both SIGHTINGS and SPOTTERS tables in full, rather than just reading the index on `spotter_id` (what used to be the primary key) and finding that value in sightings table.

In fact, with the PK and FK, it's even better because any `spotter.spotter_id` must only be in that table once (because it's a primary key), and any `spotter_id` in `sightings` must also be in `spotters` (because it's a foreign key). Once we lose these constraints, the DBMS must check each row because it can't safely stop looking once it finds one since it's not a primary key/foreign key.

The original query took 750 milliseconds to run and after dropping the primary key (and foreign key) the same query took 860 milliseconds.

5c – Index

```
ANALYZE INDEX PK_BIRD_ID VALIDATE STRUCTURE;
```

The index_stats table is populated with interesting statistics on the index as just analysed from the previous command, with the data columns.

What's the Height of the tree?

```
Select height from index_stats;
    HEIGHT
----------
         2

Elapsed: 00:00:00.01
```

How many leaf blocks are in the tree?

```
select lf_blks from index_stats;
   LF_BLKS
----------
         9

Elapsed: 00:00:00.01
```

How many block accesses are needed for a direct full scan of the BIRDS table?

Analyse it first, in some cases the user_tables is empty so have to also `gather stats` for it. I issued dbms_stats.gather_schema_stats command for my own user account.

```
ANALYZE TABLE BIRDS VALIDATE STRUCTURE;
Exec dbms_stats.gather_schema_stats('s4079161')

PL/SQL procedure successfully completed.

Elapsed: 00:00:04.43

SELECT TABLE_NAME, BLOCKS
FROM USER_TABLES;

TABLE_NAME                       BLOCKS
------------------------------ ----------
BIRDS                                  20
MV_ORGANISATION_BIRD_COUNT              5
ORGANISATIONS                           5
SIGHTINGS                            5164
SPOTTERS                               28
```

```
Elapsed: 00:00:00.01
```

So, 20 blocks accesses are needed for a direct full scan of the BIRDS table. I could have added a `where` clause to just show just the blocks of the BIRDS table, but why not see it all in context.