Oracle SQL Tuning Basics Part II

Description:
BISP is committed to provide BEST learning material to the beginners and advance learners. In the same series, we have prepared a complete end-to end Hands-on Guide SQL optimization tips. The document focuses on basic SQL optimization. See our youtube collections for more details.

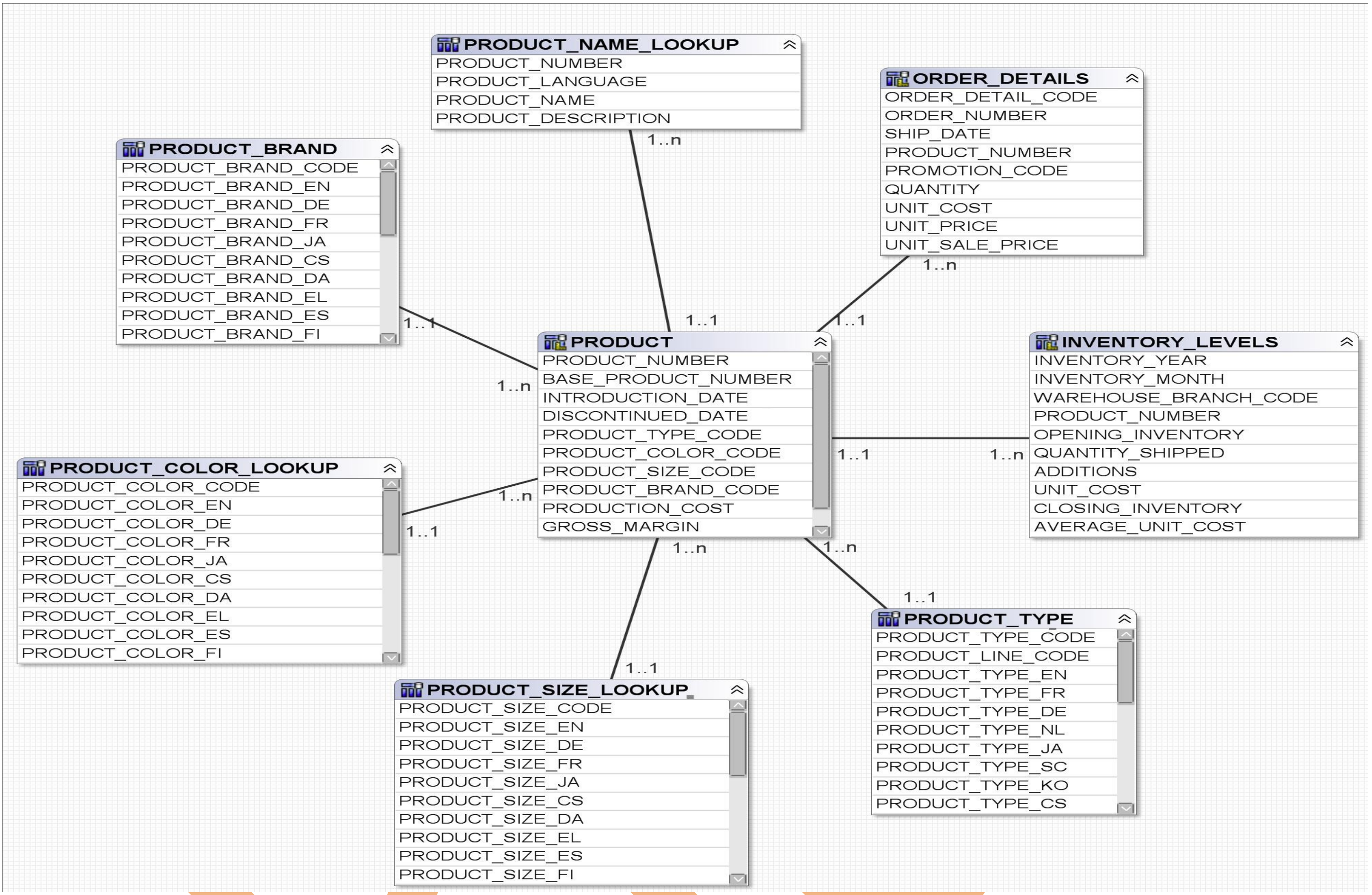
History:

Version	Description Change	Author	Publish Date
0.1	Initial Draft	Kuldeep Mishra	12th Aug 2011
0.1	Review#1	Amit Sharma	18th Aug 2011

Contents

Source Data Modeling:

We'll using these tables for all the below examples.



These indexes work very much the same way as the index in the back of this book. You build an index based on one or more columns in the table. Those column values are stored in the index. Say we create an index on the EMPLOYEE_ID column. Our index would have 500 million EMPLOYEE_ID values. Also in that index, with each EMPLOYEE_ID, is an address that tells Oracle exactly where that EMPLOYEE_ID is located in the table.

How are Indexes used?

- To quickly find specific rows by avoiding a Full Table Scan
- To avoid a table access altogether
- To avoid a sort

Oracle Indexes

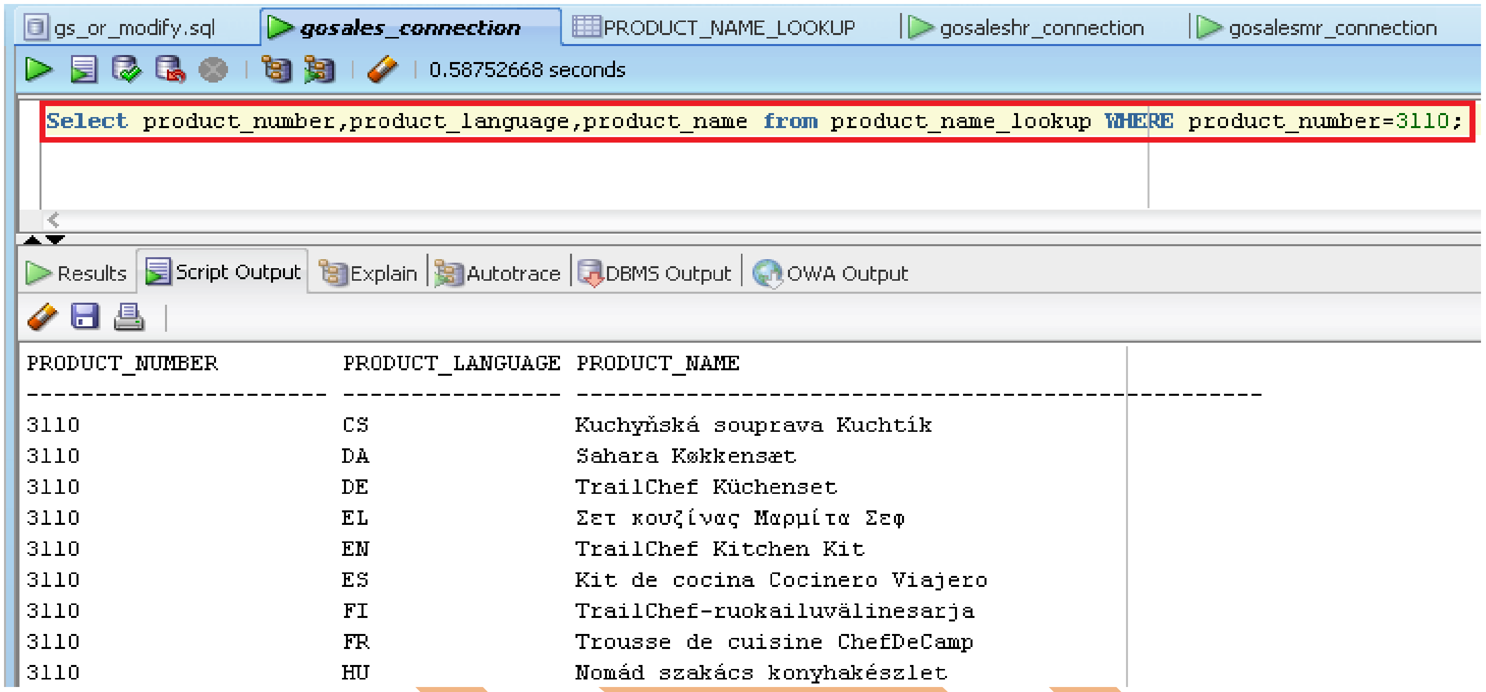
Indexes may be used for three types of conditions.

i) Equality

ii)Unbounded Range

iii)Bounded Range

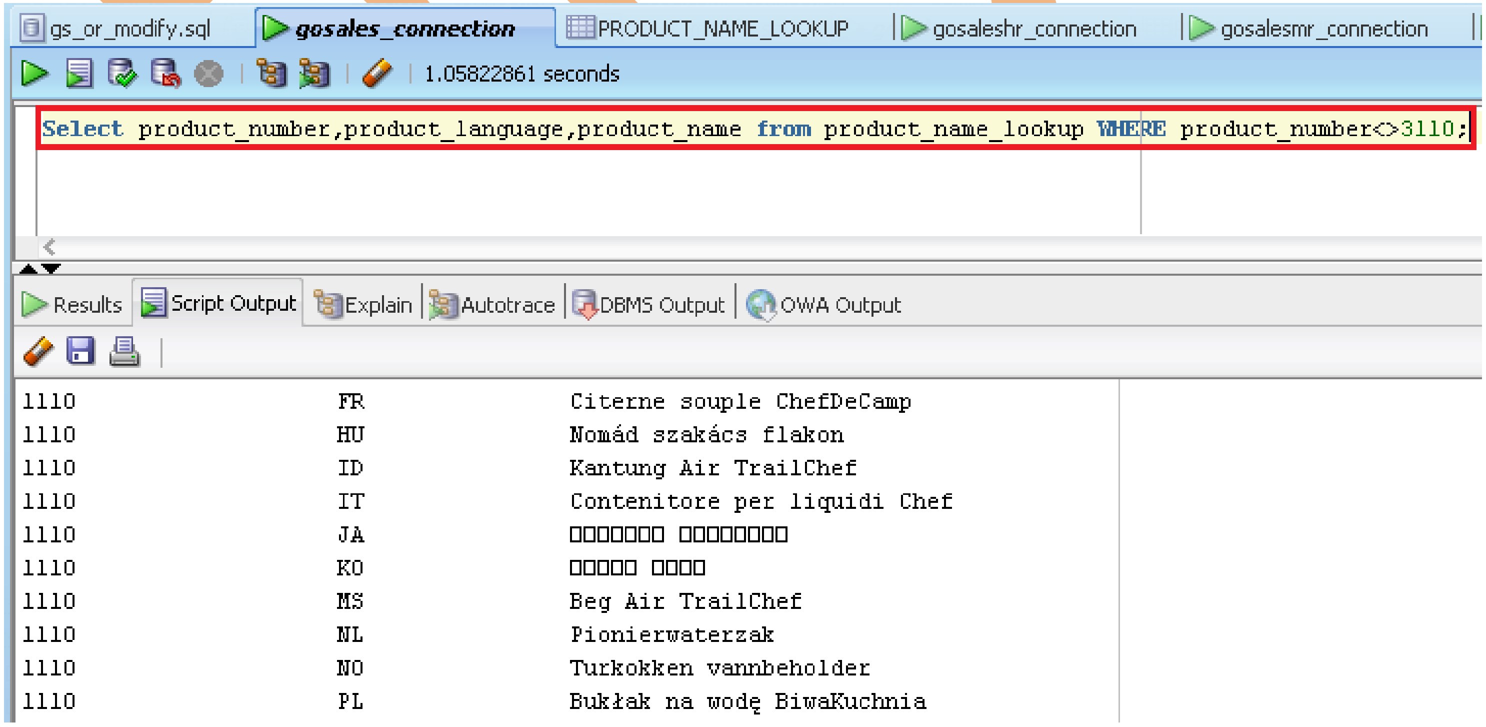
Equality
Select product_number,product_language,product_name from product_name_lookup WHERE product_number=3110;



```
Select product_number,product_language,product_name from product_name_lookup WHERE product_number=3110;
```

```
PRODUCT_NUMBER         PRODUCT_LANGUAGE PRODUCT_NAME
---------------------- ---------------- --------------------------------------------------
3110                   CS               Kuchyňská souprava Kuchtík
3110                   DA               Sahara Køkkensæt
3110                   DE               TrailChef Küchenset
3110                   EL               Σετ κουζίνας Μαρμίτα Σεφ
3110                   EN               TrailChef Kitchen Kit
3110                   ES               Kit de cocina Cocinero Viajero
3110                   FI               TrailChef-ruokailuvälinesarja
3110                   FR               Trousse de cuisine ChefDeCamp
3110                   HU               Nomád szakács konyhakészlet
```

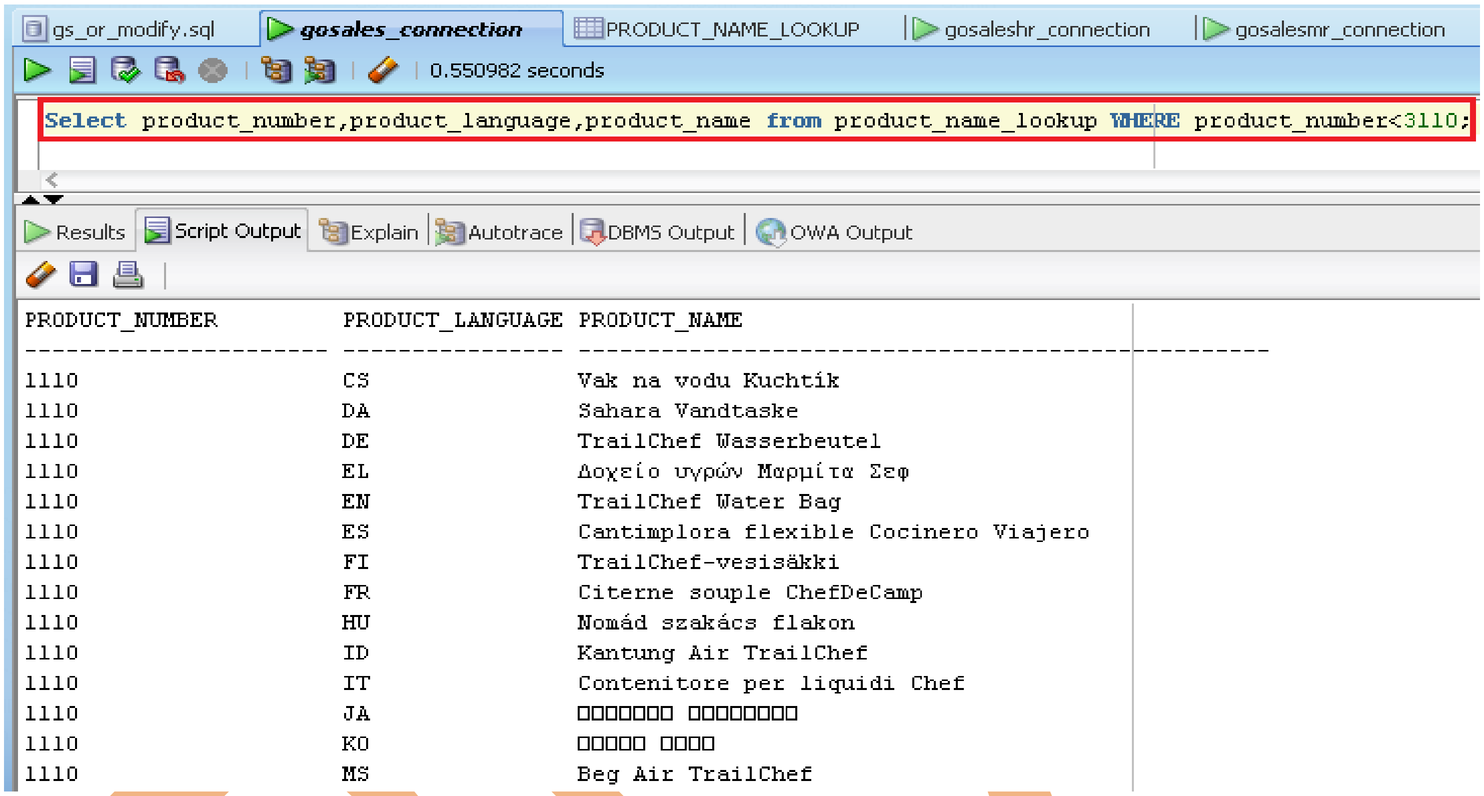
Unbounded Range
Select product_number,product_language,product_name from product_name_lookup WHERE product_number<>3110;



```
Select product_number,product_language,product_name from product_name_lookup WHERE product_number<>3110;
```

```
1110                   FR               Citerne souple ChefDeCamp
1110                   HU               Nomád szakács flakon
1110                   ID               Kantung Air TrailChef
1110                   IT               Contenitore per liquidi Chef
1110                   JA               ??????? ????????
1110                   KO               ????? ????
1110                   MS               Beg Air TrailChef
1110                   NL               Pionierwaterzak
1110                   NO               Turkokken vannbeholder
1110                   PL               Bukłak na wodę BiwaKuchnia
```

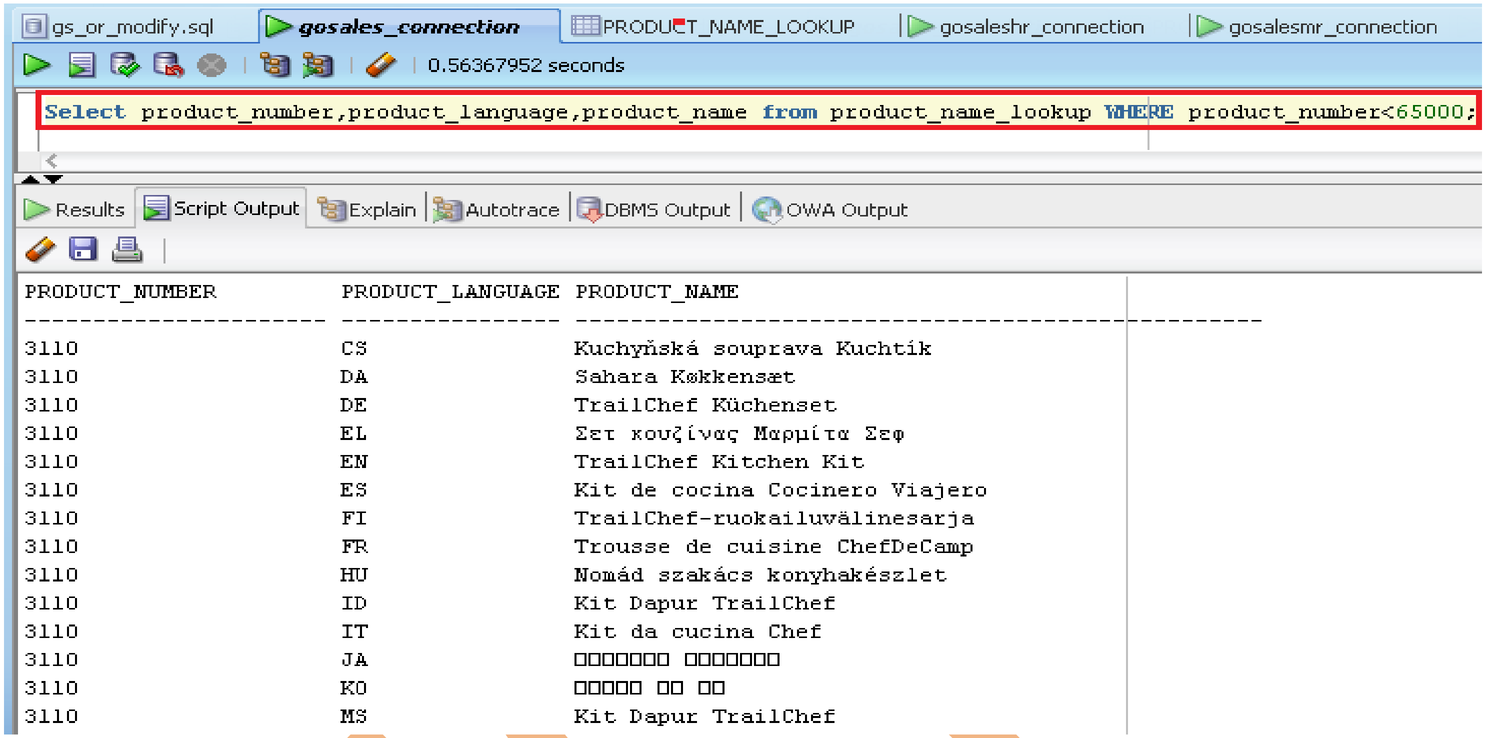
Select product_number,product_language,product_name from product_name_lookup WHERE product_number<3110;



```
Select product_number,product_language,product_name from product_name_lookup WHERE product_number<3110;
```

```
PRODUCT_NUMBER          PRODUCT_LANGUAGE PRODUCT_NAME
----------------------- ---------------- ---------------------------------------------------
1110                    CS               Vak na vodu Kuchtík
1110                    DA               Sahara Vandtaske
1110                    DE               TrailChef Wasserbeutel
1110                    EL               Δοχείο υγρών Μαρμίτα Σεφ
1110                    EN               TrailChef Water Bag
1110                    ES               Cantimplora flexible Cocinero Viajero
1110                    FI               TrailChef-vesisäkki
1110                    FR               Citerne souple ChefDeCamp
1110                    HU               Nomád szakács flakon
1110                    ID               Kantung Air TrailChef
1110                    IT               Contenitore per liquidi Chef
1110                    JA               □□□□□□□ □□□□□□□□
1110                    KO               □□□□□ □□□□
1110                    MS               Beg Air TrailChef
```

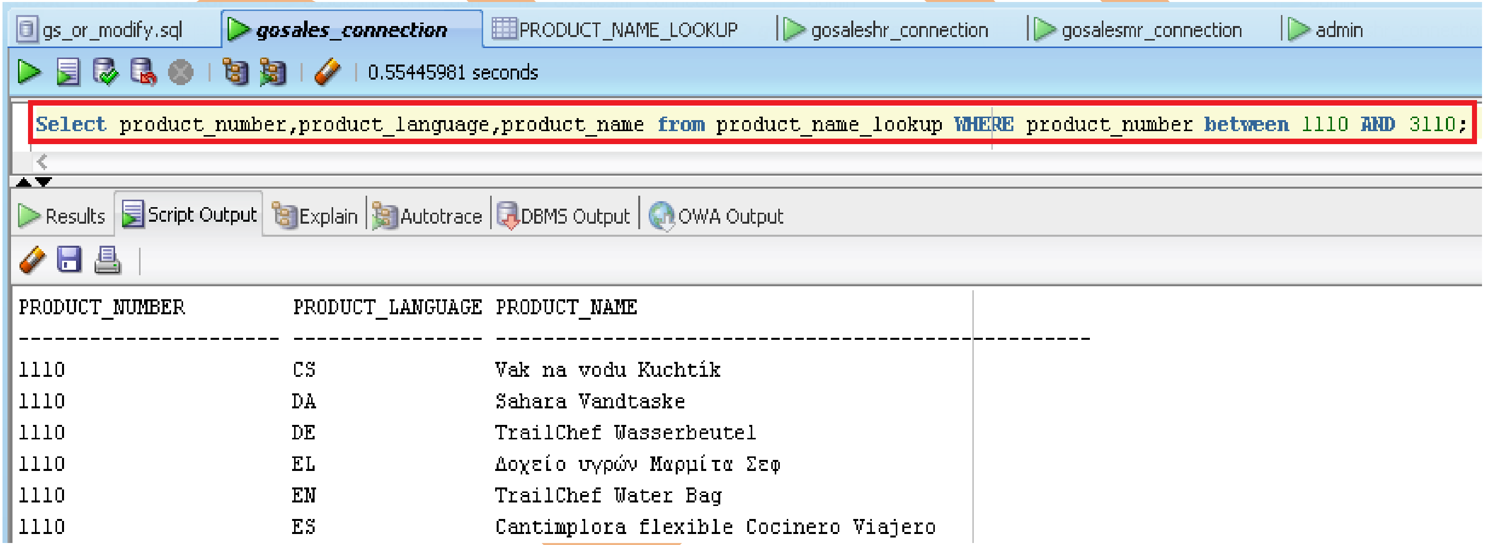
Select product_number,product_language,product_name from product_name_lookup WHERE product_number<65000;



```
Select product_number,product_language,product_name from product_name_lookup WHERE product_number<65000;
```

```
PRODUCT_NUMBER          PRODUCT_LANGUAGE PRODUCT_NAME
---------------------- ---------------- --------------------------------------------------
3110                    CS               Kuchyňská souprava Kuchtík
3110                    DA               Sahara Køkkensæt
3110                    DE               TrailChef Küchenset
3110                    EL               Σετ κουζίνας Μαρμίτα Σεφ
3110                    EN               TrailChef Kitchen Kit
3110                    ES               Kit de cocina Cocinero Viajero
3110                    FI               TrailChef-ruokailuvälinesarja
3110                    FR               Trousse de cuisine ChefDeCamp
3110                    HU               Nomád szakács konyhakészlet
3110                    ID               Kit Dapur TrailChef
3110                    IT               Kit da cucina Chef
3110                    JA               □□□□□□□ □□□□□□□
3110                    KO               □□□□□ □□ □□
3110                    MS               Kit Dapur TrailChef
```

Bounded Range
Select product_number,product_language,product_name from product_name_lookup WHERE product_number between 1110 AND 3110;



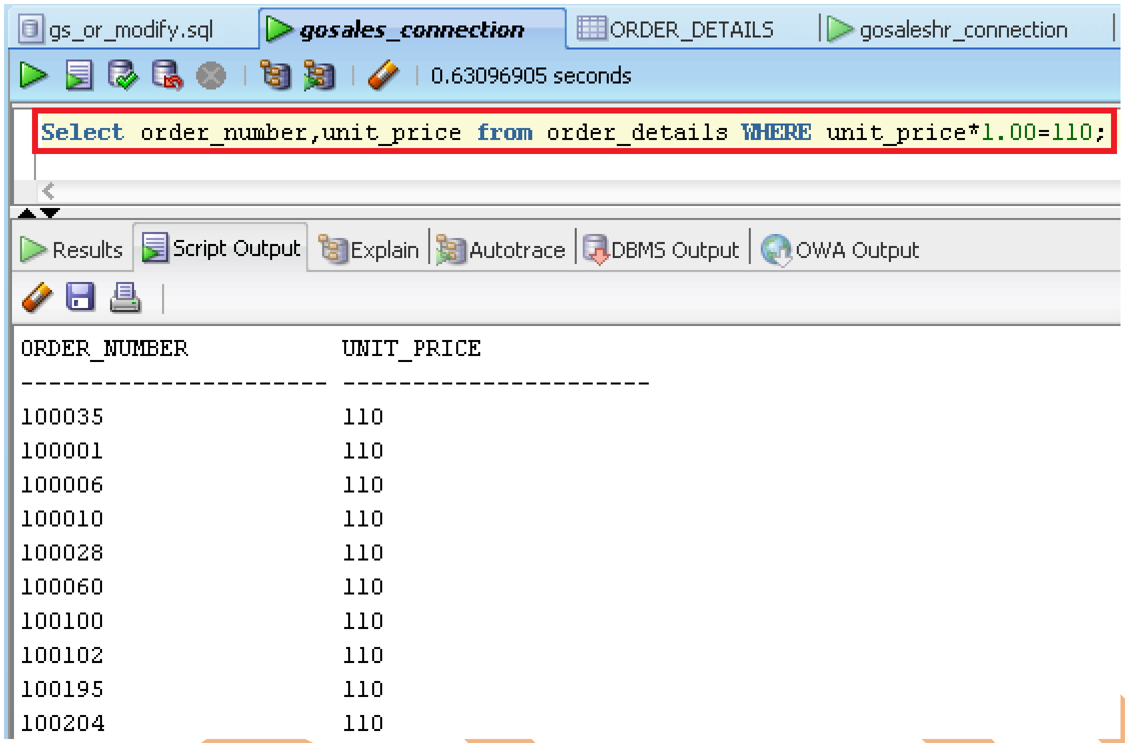
```
Select product_number,product_language,product_name from product_name_lookup WHERE product_number between 1110 AND 3110;
```

```
PRODUCT_NUMBER          PRODUCT_LANGUAGE PRODUCT_NAME
---------------------- ---------------- --------------------------------------------------
1110                    CS               Vak na vodu Kuchtík
1110                    DA               Sahara Vandtaske
1110                    DE               TrailChef Wasserbeutel
1110                    EL               Δοχείο υγρών Μαρμίτα Σεφ
1110                    EN               TrailChef Water Bag
1110                    ES               Cantimplora flexible Cocinero Viajero
```

Above examples show when the oracle optimizer can use indexes. When we want to use these three types of condition, we may be used indexes.The optimizer considers selectivity of the operation before using an index. If the Not Equal(<>) present is present then the index is not used.

Transformed Index

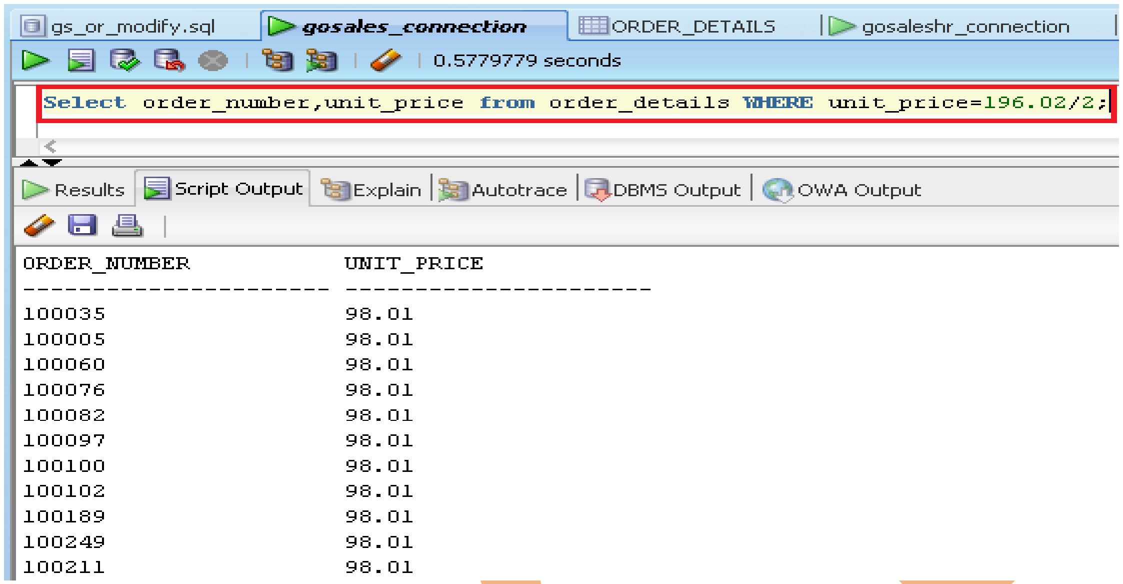
Select order_number,unit_price from order_details WHERE unit_price*1.00=110;



```
Select order_number,unit_price from order_details WHERE unit_price*1.00=110;
```

```
ORDER_NUMBER           UNIT_PRICE
---------------------- ----------------------
100035                 110
100001                 110
100006                 110
100010                 110
100028                 110
100060                 110
100100                 110
100102                 110
100195                 110
100204                 110
```

Select order_number,unit_price from order_details WHERE unit_price=196.02/2;



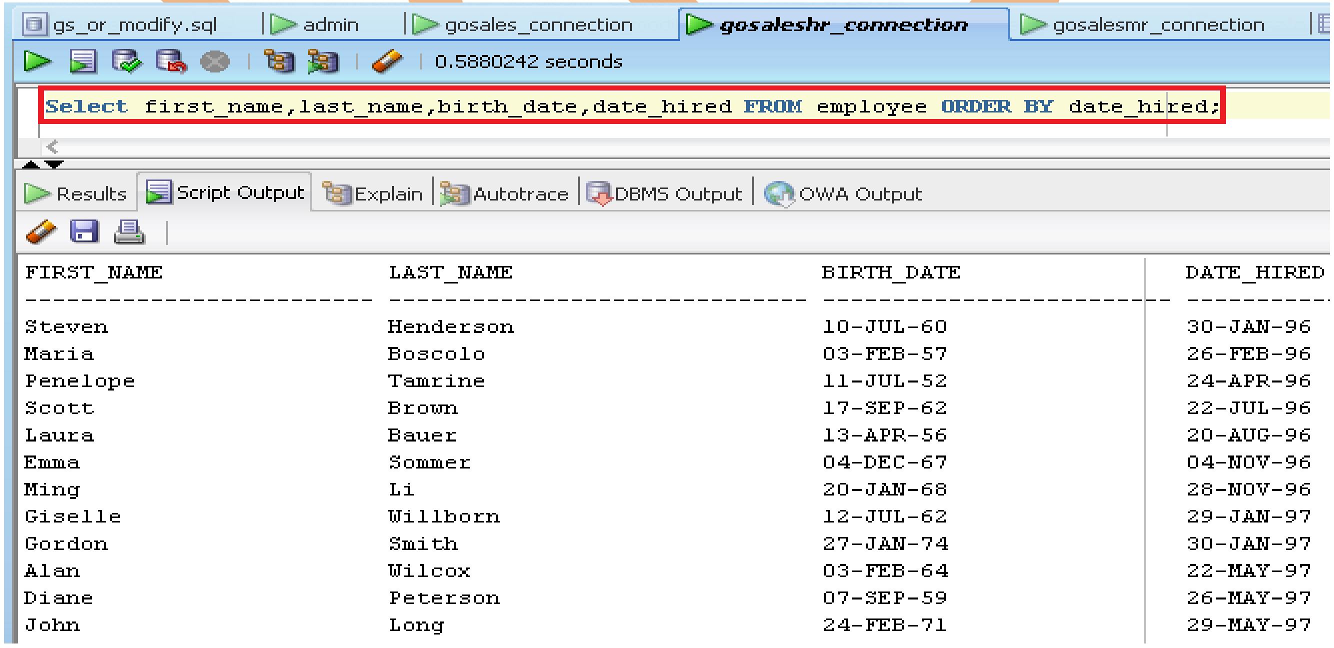
```
Select order_number,unit_price from order_details WHERE unit_price=196.02/2;
```

```
ORDER_NUMBER           UNIT_PRICE
---------------------- ----------------------
100035                 98.01
100005                 98.01
100060                 98.01
100076                 98.01
100082                 98.01
100097                 98.01
100100                 98.01
100102                 98.01
100189                 98.01
100249                 98.01
100211                 98.01
```

The result of above i and ii query show that the unit_cost column is indexed.
-If the indexed column is part of an expression in the where clause then the (i) query can happen.
-If the index column appears clean in the where clause and even then may be used based only on selectivity then only an index may be usable.

Tune the ORDER BY Clause

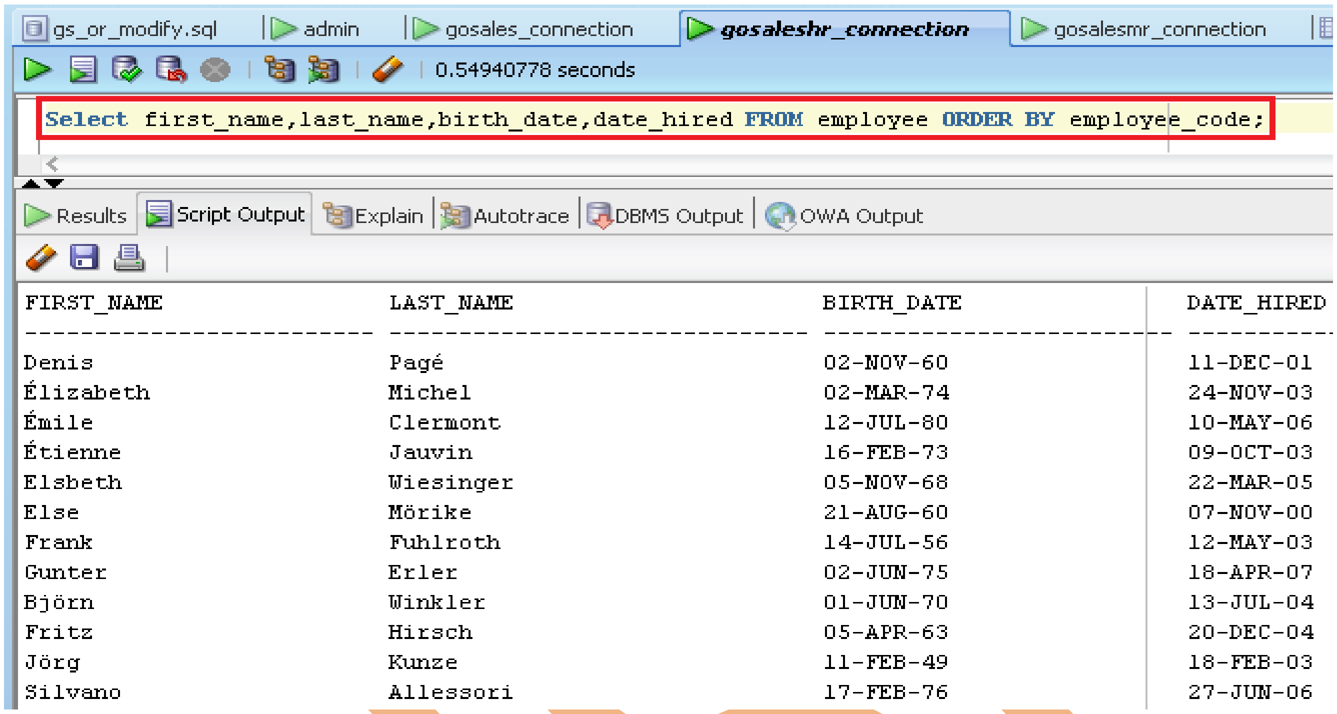
i) Select first_name,last_name,birth_date,date_hired FROM employee ORDER BY date_hired;



```
Select first_name,last_name,birth_date,date_hired FROM employee ORDER BY date_hired;
```

```
FIRST_NAME                LAST_NAME                      BIRTH_DATE                DATE_HIRED
------------------------- ------------------------------ ------------------------- ----------
Steven                    Henderson                      10-JUL-60                 30-JAN-96
Maria                     Boscolo                        03-FEB-57                 26-FEB-96
Penelope                  Tamrine                        11-JUL-52                 24-APR-96
Scott                     Brown                          17-SEP-62                 22-JUL-96
Laura                     Bauer                          13-APR-56                 20-AUG-96
Emma                      Sommer                         04-DEC-67                 04-NOV-96
Ming                      Li                             20-JAN-68                 28-NOV-96
Giselle                   Willborn                       12-JUL-62                 29-JAN-97
Gordon                    Smith                          27-JAN-74                 30-JAN-97
Alan                      Wilcox                         03-FEB-64                 22-MAY-97
Diane                     Peterson                       07-SEP-59                 26-MAY-97
John                      Long                           24-FEB-71                 29-MAY-97
```

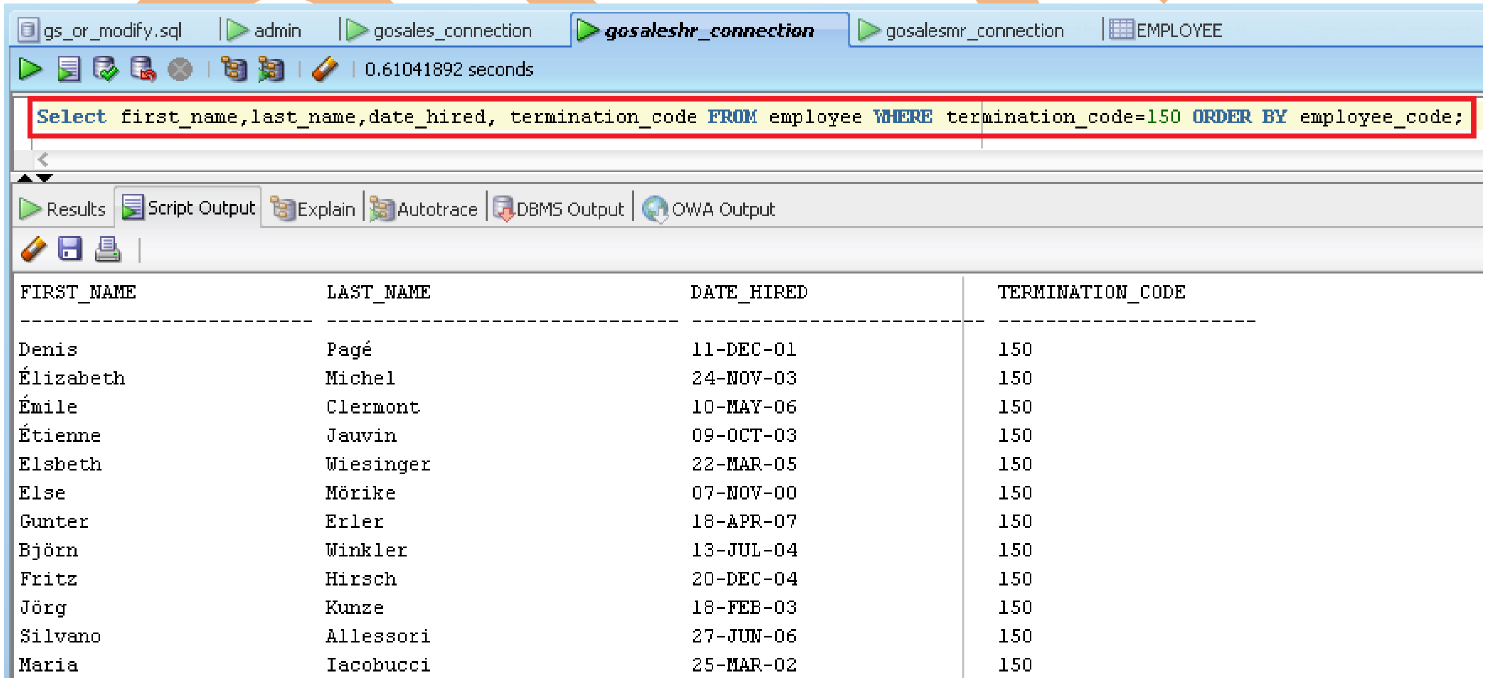
Select first_name,last_name,birth_date,date_hired FROM employee ORDER BY employee_code;



```
Select first_name,last_name,birth_date,date_hired FROM employee ORDER BY employee_code;
```

```
FIRST_NAME                 LAST_NAME                      BIRTH_DATE                DATE_HIRED
-------------------------- ------------------------------ ------------------------- -----------
Denis                      Pagé                           02-NOV-60                 11-DEC-01
Élizabeth                  Michel                         02-MAR-74                 24-NOV-03
Émile                      Clermont                       12-JUL-80                 10-MAY-06
Étienne                    Jauvin                         16-FEB-73                 09-OCT-03
Elsbeth                    Wiesinger                      05-NOV-68                 22-MAR-05
Else                       Mörike                         21-AUG-60                 07-NOV-00
Frank                      Fuhlroth                       14-JUL-56                 12-MAY-03
Gunter                     Erler                          02-JUN-75                 18-APR-07
Björn                      Winkler                        01-JUN-70                 13-JUL-04
Fritz                      Hirsch                         05-APR-63                 20-DEC-04
Jörg                       Kunze                          11-FEB-49                 18-FEB-03
Silvano                    Allessori                      17-FEB-76                 27-JUN-06
```

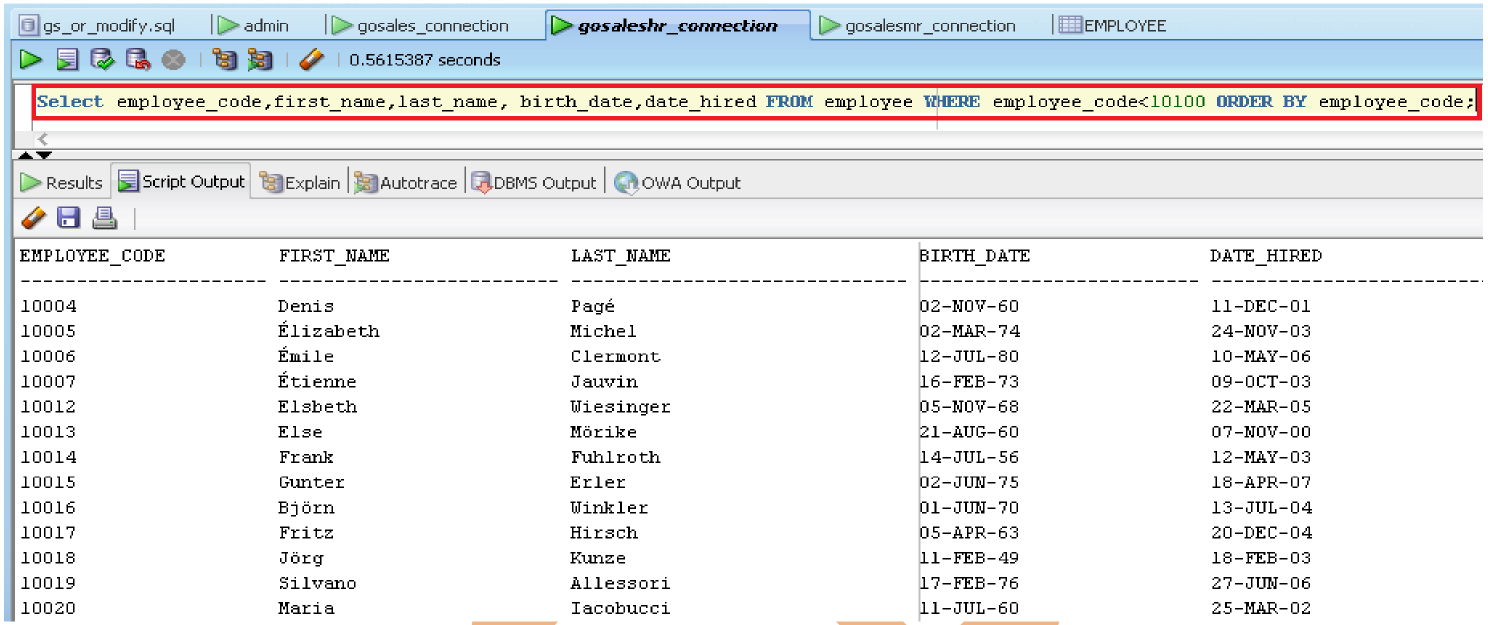
Select first_name,last_name,termination_code FROM employee WHERE termination_code =150 ORDER BY employee_code;



```
Select first_name,last_name,date_hired, termination_code FROM employee WHERE termination_code=150 ORDER BY employee_code;
```

```
FIRST_NAME                LAST_NAME                      DATE_HIRED                TERMINATION_CODE
------------------------- ------------------------------ ------------------------- ----------------------
Denis                     Pagé                           11-DEC-01                 150
Élizabeth                 Michel                         24-NOV-03                 150
Émile                     Clermont                       10-MAY-06                 150
Étienne                   Jauvin                         09-OCT-03                 150
Elsbeth                   Wiesinger                      22-MAR-05                 150
Else                      Mörike                         07-NOV-00                 150
Gunter                    Erler                          18-APR-07                 150
Björn                     Winkler                        13-JUL-04                 150
Fritz                     Hirsch                         20-DEC-04                 150
Jörg                      Kunze                          18-FEB-03                 150
Silvano                   Allessori                      27-JUN-06                 150
Maria                     Iacobucci                      25-MAR-02                 150
```

Select employee_code ,first_name,last_name,birth_date,date_hired FROM employee WHERE   employee_code<10100 ORDER BY employee_code;



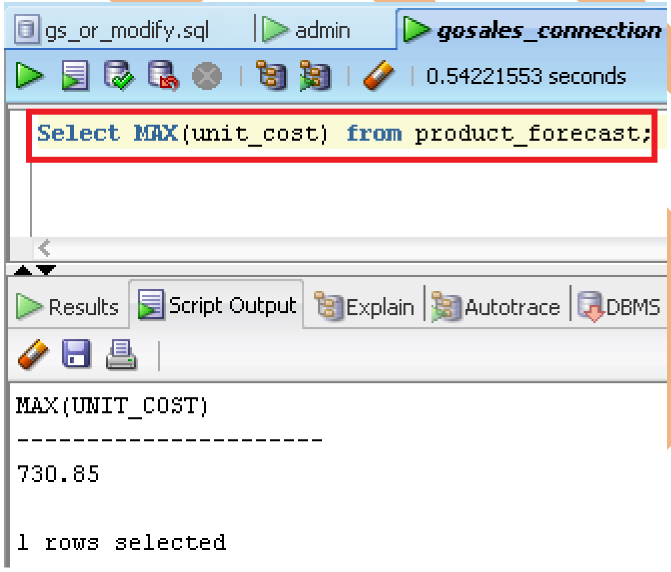
```
Select employee_code,first_name,last_name, birth_date,date_hired FROM employee WHERE employee_code<10100 ORDER BY employee_code;
```

```
EMPLOYEE_CODE          FIRST_NAME               LAST_NAME                      BIRTH_DATE                DATE_HIRED
---------------------- ------------------------ ------------------------------ ------------------------- -------------------------
10004                  Denis                    Pagé                           02-NOV-60                 11-DEC-01
10005                  Élizabeth                Michel                         02-MAR-74                 24-NOV-03
10006                  Émile                    Clermont                       12-JUL-80                 10-MAY-06
10007                  Étienne                  Jauvin                         16-FEB-73                 09-OCT-03
10012                  Elsbeth                  Wiesinger                      05-NOV-68                 22-MAR-05
10013                  Else                     Mörike                         21-AUG-60                 07-NOV-00
10014                  Frank                    Fuhlroth                       14-JUL-56                 12-MAY-03
10015                  Gunter                   Erler                          02-JUN-75                 18-APR-07
10016                  Björn                    Winkler                        01-JUN-70                 13-JUL-04
10017                  Fritz                    Hirsch                         05-APR-63                 20-DEC-04
10018                  Jörg                     Kunze                          11-FEB-49                 18-FEB-03
10019                  Silvano                  Allessori                      17-FEB-76                 27-JUN-06
10020                  Maria                    Iacobucci                      11-JUL-60                 25-MAR-02
```

If Oracle server is performing all the sort activity in Program Global Area, the performance is acceptable. Sometimes intermediate results write to disk by oracle server. By using various tools we can find the statistics on the sort operation. Sort operations caused by the order by clause show by the query(i-iii).Possible ways to tune the order by clause -->By tuning PGA memory or By creating Indexes.

Retrieve a MAX value

Example#1 Select MAX(unit_cost) from product_forecast;

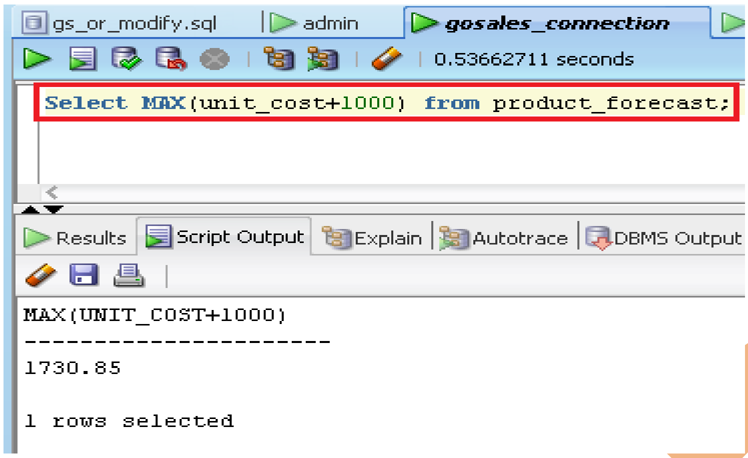


```
Select MAX(unit_cost) from product_forecast;
```

```
MAX(UNIT_COST)
----------------------
730.85

1 rows selected
```

Example#2 Select MAX(unit_cost+1000) from product_forecast;

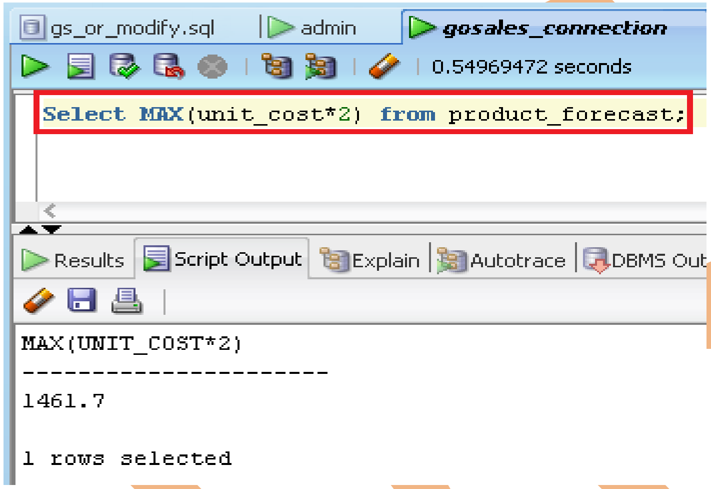


```
Select MAX(unit_cost+1000) from product_forecast;
```

```
MAX(UNIT_COST+1000)
----------------------
1730.85

1 rows selected
```

Example#3 Select MAX(unit_cost*2) from product_forecast;



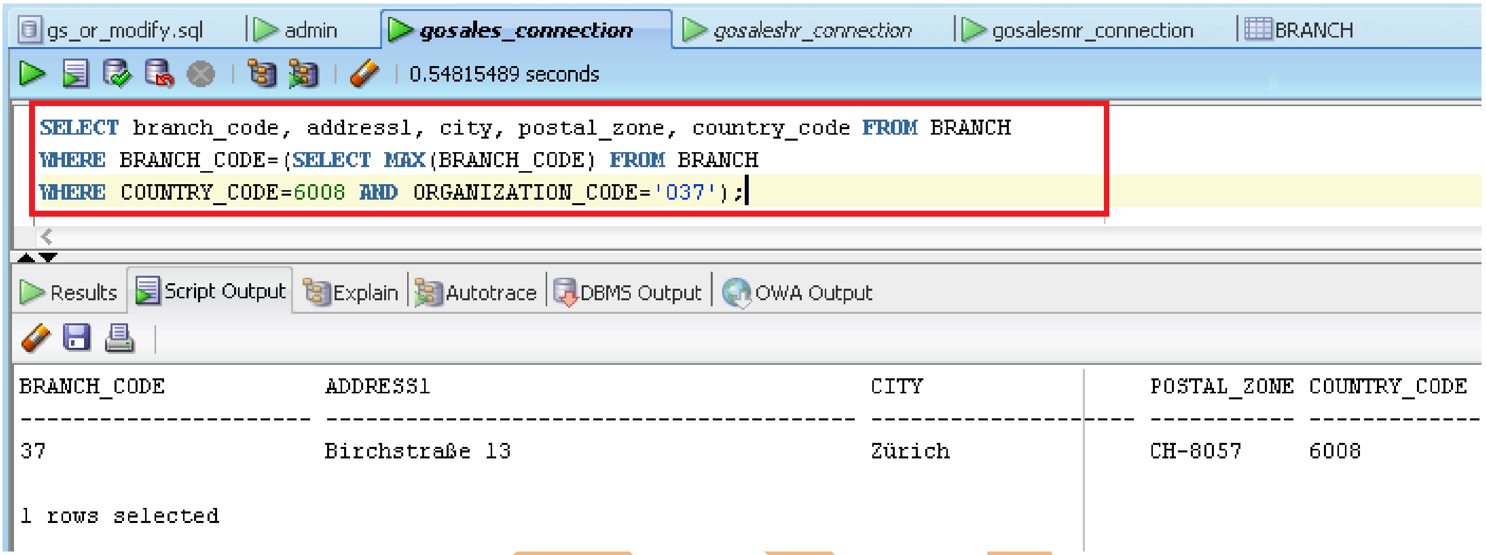
```
Select MAX(unit_cost*2) from product_forecast;
```

```
MAX(UNIT_COST*2)
----------------------
1461.7

1 rows selected
```

An Index can be useful to retrieve a maximum value(and a minimum value) is shown by the first two queries (i)(ii).The optimizer must scan the full table and perform sort to find maximum and minimum value if no index is available. Operation on the indexed column value prevents the index being used in the query (iii).

Example#4
SELECT branch_code, address1, city, postal_zone, country_code FROM BRANCH
WHERE BRANCH_CODE=(SELECT MAX(BRANCH_CODE) FROM BRANCH
WHERE COUNTRY_CODE=6008 AND ORGANIZATION_CODE='037');



```
SELECT branch_code, address1, city, postal_zone, country_code FROM BRANCH
WHERE BRANCH_CODE=(SELECT MAX(BRANCH_CODE) FROM BRANCH
WHERE COUNTRY_CODE=6008 AND ORGANIZATION_CODE='037');
```

```
BRANCH_CODE            ADDRESS1                                 CITY                  POSTAL_ZONE COUNTRY_CODE
---------------------- ---------------------------------------- --------------------- ----------- ------------
37                     Birchstraße 13                           Zürich                CH-8057     6008

1 rows selected
```

In this above query the subquery executes before the main query and result of subquery is used by the main query.

Correlated Subquery

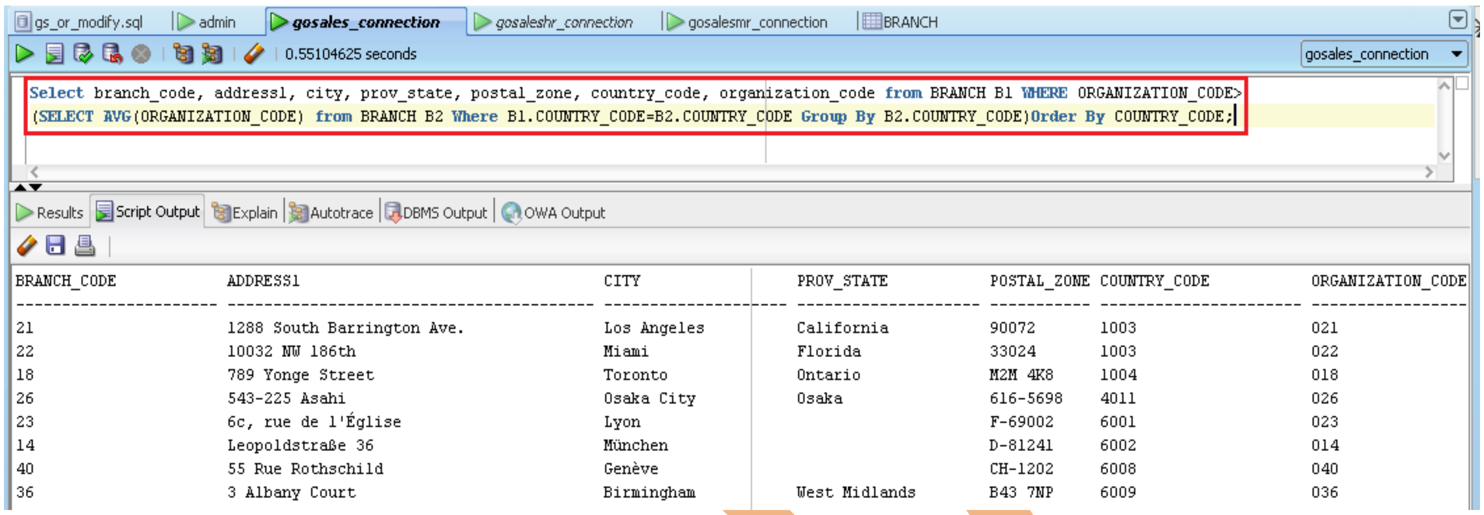
Correlated subqueries and slow because the sub-query is executed ONCE for each row returned by the outer query. In a correlated subquery, the inner query uses information from the outer query and executes once for every row in the outer query. This correlation is accomplished by using a reference to the outside query within the subquery. The use of a correlated subquery is not very efficient. Using joins rather than a correlated subquery enables the optimizer to determine how to correlate the data in the most efficient way. A practical use of a correlated subquery is to transfer data from one table to another.

Starting in Oracle9i release 2 we see an incorporation of the SQL-99 WITH clause, a tool for materializing subqueries. Oracle offers three types of materialization, each with its own type and duration:

Select branch_code, address1, city, prov_state, postal_zone, country_code, organization_code from BRANCH B1 WHERE ORGANIZATION_CODE> (SELECT AVG(ORGANIZATION_CODE) from BRANCH B2 Where B1.COUNTRY_CODE=B2.COUNTRY_CODE Group By B2.COUNTRY_CODE)Order By COUNTRY_CODE;

Rewrite the above correlated query using with clause

With cr_exam as
(SELECT AVG(ORGANIZATION_CODE) from BRANCH B2 Where
B1.COUNTRY_CODE=B2.COUNTRY_CODE
Group By B2.COUNTRY_CODE)
Select cr_exam.branch_code, cr_exam.address1, cr_exam.city, cr_exam.prov_state, cr_exam.postal_zone, cr_exam.country_code, cr_exam.organization_code from BRANCH B1, cr_exam WHERE B1.ORGANIZATION_CODE>cr_exam. ORGANIZATION_CODE



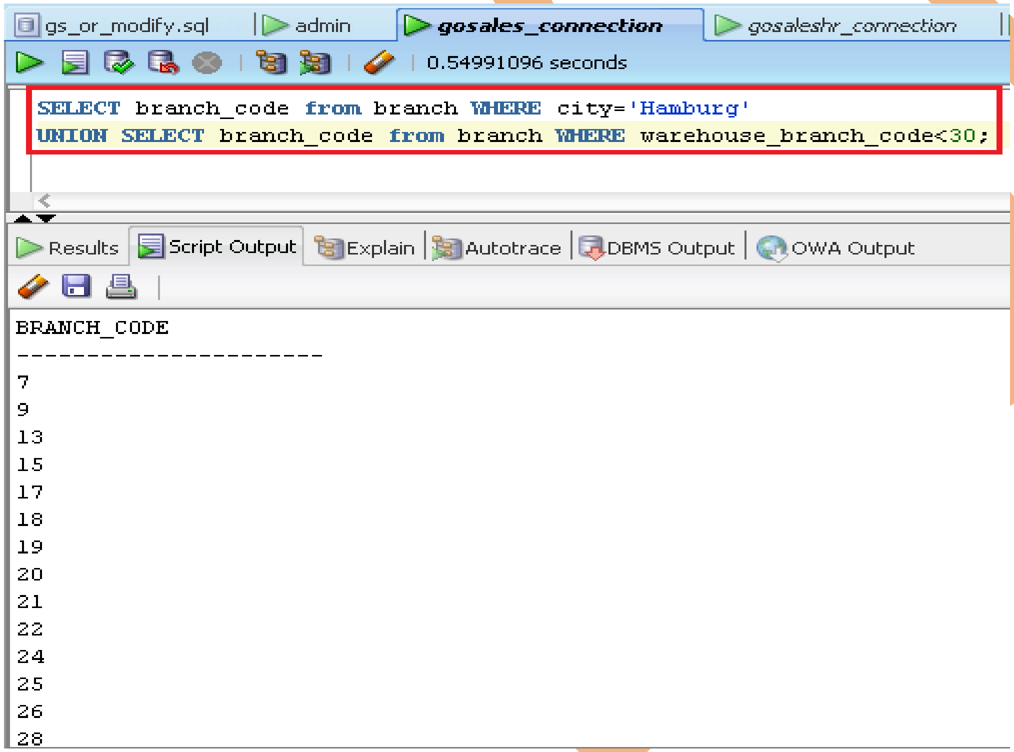
```
Select branch_code, address1, city, prov_state, postal_zone, country_code, organization_code from BRANCH B1 WHERE ORGANIZATION_CODE>
(SELECT AVG(ORGANIZATION_CODE) from BRANCH B2 Where B1.COUNTRY_CODE=B2.COUNTRY_CODE Group By B2.COUNTRY_CODE)Order By COUNTRY_CODE;
```

```
BRANCH_CODE           ADDRESS1                                 CITY                 PROV_STATE           POSTAL_ZONE COUNTRY_CODE          ORGANIZATION_CODE
--------------------- ---------------------------------------- -------------------- -------------------- ----------- --------------------- -----------------
21                    1288 South Barrington Ave.               Los Angeles          California           90072       1003                  021
22                    10032 NW 186th                           Miami                Florida              33024       1003                  022
18                    789 Yonge Street                         Toronto              Ontario              M2M 4K8     1004                  018
26                    543-225 Asahi                            Osaka City           Osaka                616-5698    4011                  026
23                    6c, rue de l'Église                      Lyon                                      F-69002     6001                  023
14                    Leopoldstraße 36                         München                                   D-81241     6002                  014
40                    55 Rue Rothschild                        Genève                                    CH-1202     6008                  040
36                    3 Albany Court                           Birmingham           West Midlands        B43 7NP     6009                  036
```

The above query is returning the data about branch.

Union and Union All

SELECT branch_code from branch WHERE city='Hamburg'
UNION SELECT branch_code from branch WHERE warehouse_branch_code<30;



```
SELECT branch_code from branch WHERE city='Hamburg'
UNION SELECT branch_code from branch WHERE warehouse_branch_code<30;
```

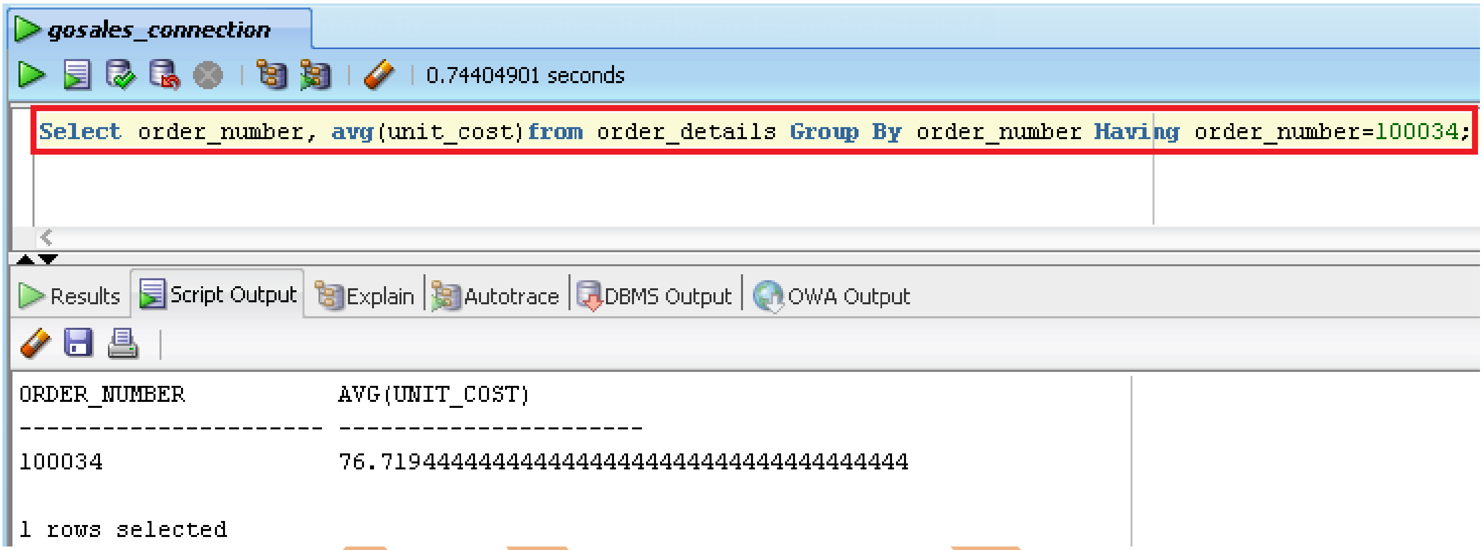
```
BRANCH_CODE
----------------------
7
9
13
15
17
18
19
20
21
22
24
25
26
28
```

In sort operations UNION operator unconditionally results. Regardless of the presence of indexes. The Sql set operators are used to remove duplicate rows that are why sorts are needed.

Rather Than: The Union All operator neither perform sort nor remove duplicate rows. We can use union all operator, when we sure about the data that there is no duplicate rows**.**

Avoid Using Having

Select order_number, avg(unit_cost)from order_details Group By order_number Having order_number=100034;



```
Select order_number, avg(unit_cost)from order_details Group By order_number Having order_number=100034;
```

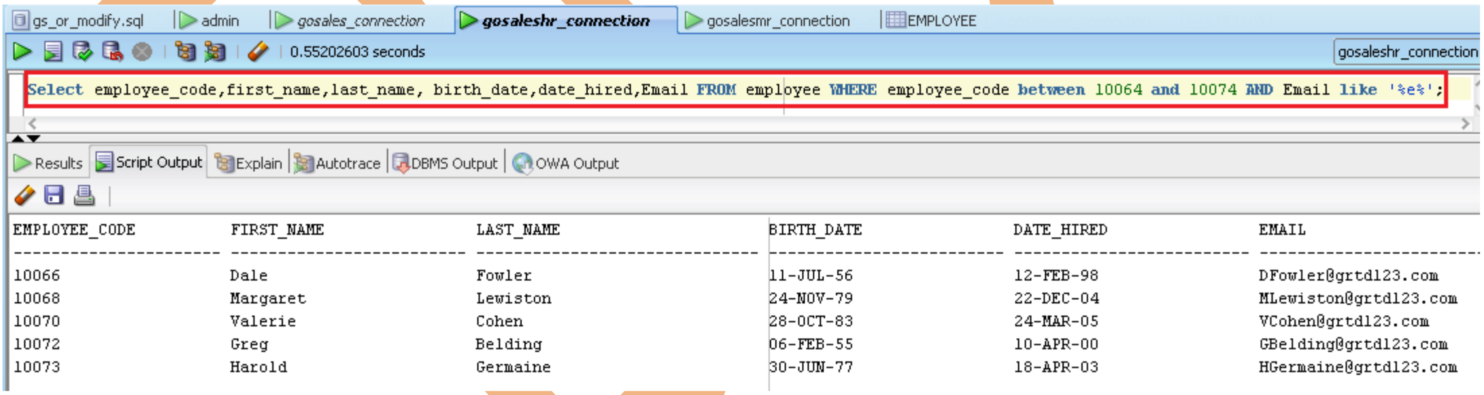
```
ORDER_NUMBER           AVG(UNIT_COST)
---------------------- ----------------------
100034                 76.7194444444444444444444444444444444444

1 rows selected
```

Above query examines the Having operator

Tune the Between Operator

Select employee_code,first_name,last_name, birth_date,date_hired,Email FROM employee WHERE employee_code between 10064 and 10074 AND Email like '%e%';



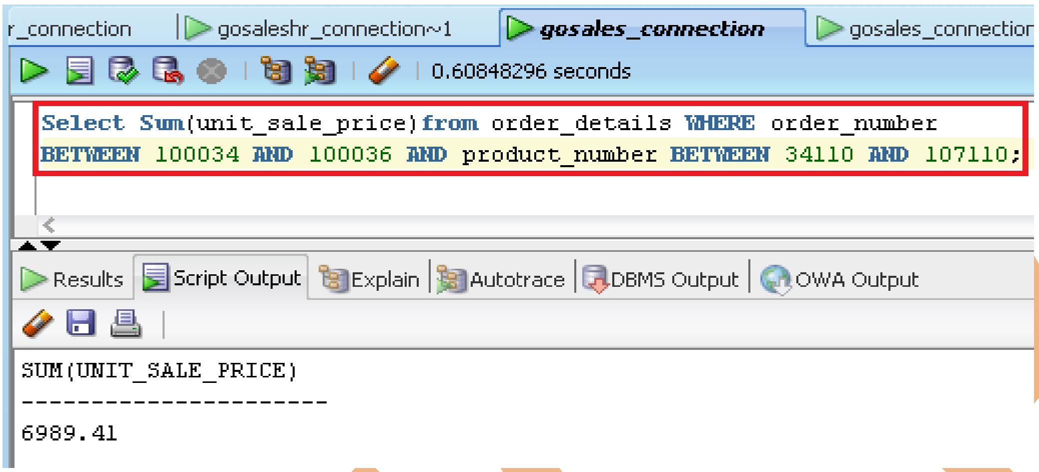
```
Select employee_code,first_name,last_name, birth_date,date_hired,Email FROM employee WHERE employee_code between 10064 and 10074 AND Email like '%e%';
```

```
EMPLOYEE_CODE          FIRST_NAME               LAST_NAME                     BIRTH_DATE                DATE_HIRED                EMAIL
---------------------- ------------------------ ----------------------------- ------------------------- ------------------------- ------------------------
10066                  Dale                     Fowler                        11-JUL-56                 12-FEB-98                 DFowler@grtd123.com
10068                  Margaret                 Lewiston                      24-NOV-79                 22-DEC-04                 MLewiston@grtd123.com
10070                  Valerie                  Cohen                         28-OCT-83                 24-MAR-05                 VCohen@grtd123.com
10072                  Greg                     Belding                       06-FEB-55                 10-APR-00                 GBelding@grtd123.com
10073                  Harold                   Germaine                      30-JUN-77                 18-APR-03                 HGermaine@grtd123.com
```

The between operator is used to evaluates whether a value lies in a specified range . for example: Empid Between 101 to 105 returns the same value as (Empid>=101) AND (Empid>=101). If Empid column is indexed and 'Empid Betwween 101 And 105' is restrictive.The optimizer might chose the index. In above query the full table scan creates by the optimizer instead. Because the condition with Between operator returns all matching rows and save the optimizer from using index scan.

Tune the Star Query by using the join Operation

Select Sum(unit_sale_price) from order_details
WHERE order_number BETWEEN 100034 AND 100036
AND
product_number BETWEEN 34110 AND 107110;



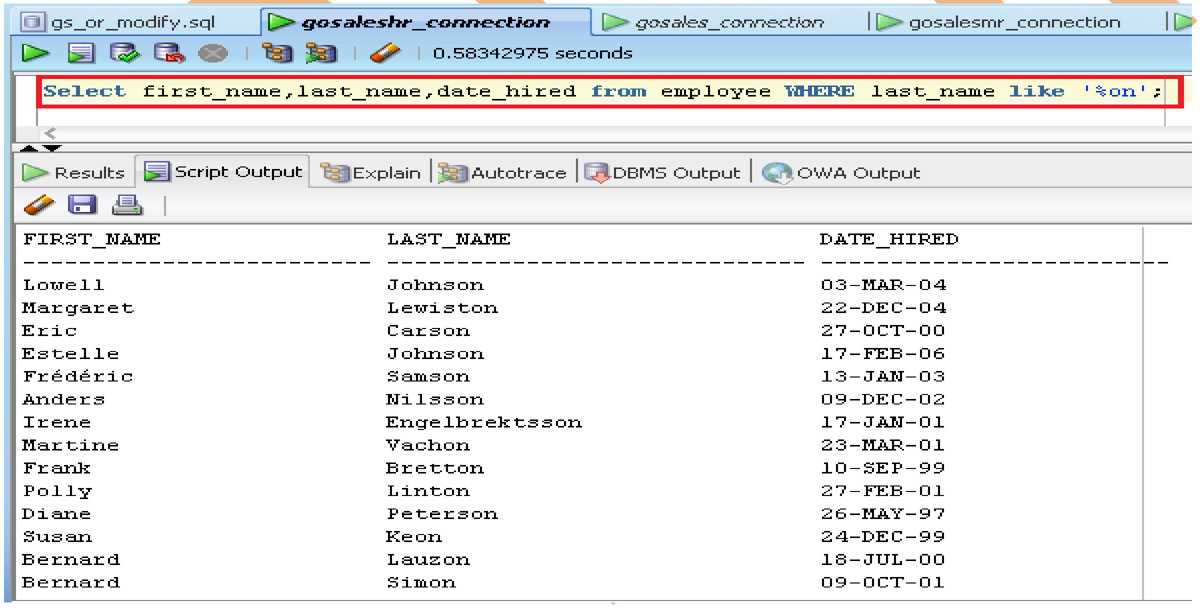
```
Select Sum(unit_sale_price)from order_details WHERE order_number
BETWEEN 100034 AND 100036 AND product_number BETWEEN 34110 AND 107110;
```

```
SUM(UNIT_SALE_PRICE)
----------------------
6989.41
```

In this above query the optimizer selected the indexed column to return the business data.

Index for Like '%string'

Select first_name,last_name,date_hired from employee WHERE last_name like '%on';



```
Select first_name,last_name,date_hired from employee WHERE last_name like '%on';
```

```
FIRST_NAME                 LAST_NAME                       DATE_HIRED
-------------------------- ------------------------------- --------------------------
Lowell                     Johnson                         03-MAR-04
Margaret                   Lewiston                        22-DEC-04
Eric                       Carson                          27-OCT-00
Estelle                    Johnson                         17-FEB-06
Frédéric                   Samson                          13-JAN-03
Anders                     Nilsson                         09-DEC-02
Irene                      Engelbrektsson                  17-JAN-01
Martine                    Vachon                          23-MAR-01
Frank                      Bretton                         10-SEP-99
Polly                      Linton                          27-FEB-01
Diane                      Peterson                        26-MAY-97
Susan                      Keon                            24-DEC-99
Bernard                    Lauzon                          18-JUL-00
Bernard                    Simon                           09-OCT-01
```

We can use index when the search pattern look as '%String%' and the index column is very selective. In the above query the search pattern start with wildcard to find employee who have their last_name ending 'on' .