Database Design – Lapdock

# Final Project for Advanced Database Management System | Group 4 | ISM6218.003F21.96801 Advanced Database Management

Text

Description automatically generated

**Team Members**

Prashanth Kumar Gajula Anisetty

Nitish Lonka

Pranay Metuku

Shiva Sai Ganesh

Lahari Goshika

# Summary of Contents

|  |  |  |
| --- | --- | --- |
| **Serial No.** | **Title** | **Page No.** |
| 1. | Purpose | **4** |
| 2. | Narrative | **4** |
| 3. | Entities to be Tracked | **4** |
| 4. | Entities with Attributes Nested | **4** |
| 5. | Business Rules | **6** |
| 6. | Entity Relationship Documentation (ERD) | **7** |
| 7. | Table Views | **8** |
| 8. | Data Synthesis | **11** |
| 9. | Data Integrity | **12** |
| 10. | Queries & Stored Proc | **13** |
| 11. | Performance Tuning | **17** |
| 13. | Data Visualization | **31** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Topic Area** | **Description** | **Group Member** | **Weight** |
| **Database Design** | This part should include a logical database design (for the relational model), using normalization to control redundancy and integrity  constraints for data quality. | Nitish  Pranay Metuku | 25% |
| **Query Writing** | This part is another chance to write SQL queries, explore transactions, and even do some database programming for stored  procedures. | Pranay Metuku  Prashanth Kumar Anisetty  Lahari Goshika | 25% |
| **Performance Tuning** | In this section, you can capitalize and extend your prior experiments with indexing, optimizer modes, partitioning, parallel execution and any other techniques you want to further  explore. | Prashanth Kumar Anisetty  Nitish  Shiva Sai | 25% |
| **Data Visualization** | Here you are free to explore any other topics of interest. Suggestions include | Shiva Sai Ganesh  Lahari Goshika | 25% |
| DBA scripts, database security, |
| interface design, data |
| visualization, data mining, and |
| NoSQL databases. |

**Purpose**

This document outlines the steps that were taken to construct a database system. It describes the items and characteristics that have been monitored in the database in detail. This paper also covers database integrity management, data generation and loading, query writing to obtain data, performance tuning using indexes, and parallelism. In this document, additional areas such as data visualization using R were also presented.

# Narrative

Customers, corporations are all involved in a warehouse like Lapdock. It is critical to digitize information today for better strategic data usage, faster administration of data organization, and management of customer-company relationships. The advantages are numerous.

There are a lot of customers at the Lapdock. When a customer opens an account with it, they are given a unique customer ID. Additional information on each client, such as their complete name, phone number, email id and address.

# Entities Identified to be Tracked

* Companies
* Category companies
* Device categories
* Model
* Configurations
* User ratings
* User table
* Order
* Items

# Entities with Attributes Nested

**Companies**

* Company ID
* Company Name

**Category companies**

* Category company ID
* Company ID
* Device Category ID

**Device Categories**

* Device Category ID
* Device category name

**Models**

* Model ID
* Model Name
* Category company ID
* Average\_rating

**Configurations**

* Configuration ID
* RAM
* Disk Storage
* Weight
* Battery Life
* Price
* Model ID
* Display
* OS
* Warranty

**User Ratings**

* Rating ID
* User rating
* User ID
* Model ID

**Users Table**

* User ID
* User Name
* Address
* User email
* Phone number

**Orders**

* Order ID
* User ID
* Order Total

**Items**

* Item ID
* Serial Number
* Configuration ID
* Order ID

# Business Rules

# Every company has wide number of device categories.

# Each model has multiple configurations and multiple user ratings

# A user can have multiple orders and give single rating to the respective item.

# Each item has unique serial number.

# Entity Relationship Diagram representing Database Design:

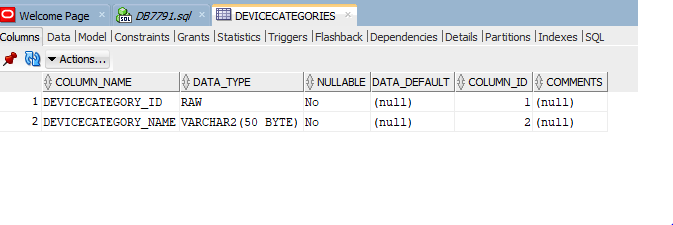
# Diagram Description automatically generated

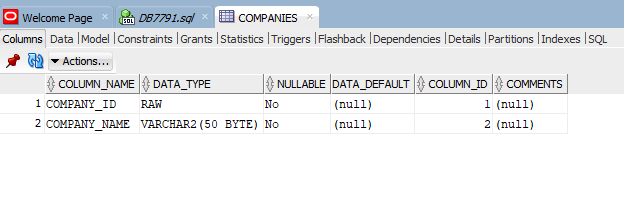
**Table Views**

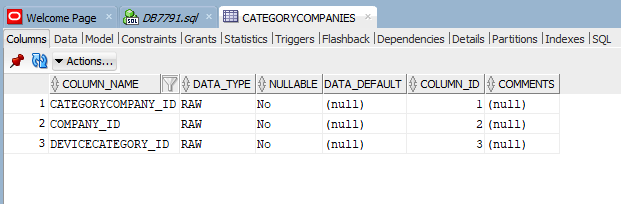
**Schema\_name: DB769 >Lapdock database tables**

**Schema\_name: DB779>Views and performance tuning queries**

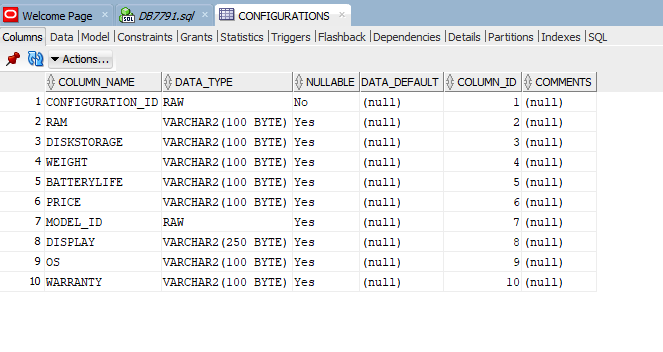
**Device Categories:** This entity stores the various categories of devices like mobiles,tablets,laptops etc available in the warehouse.

**Company :** This entity stores the names of products owned by companies.

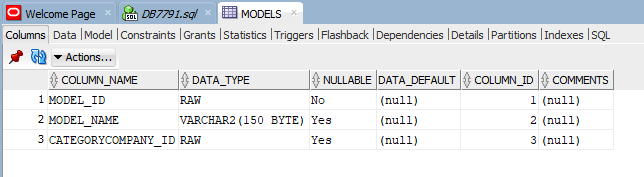
**Category Companies**: It is an associate entity between Device Categories and Company entities



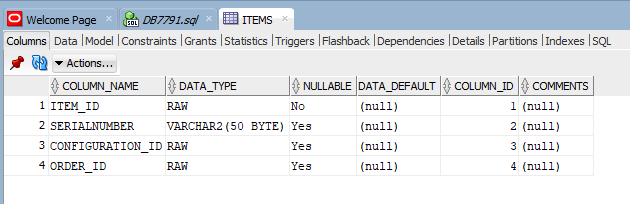
**Configuration table:** It contains the product related configuration information.



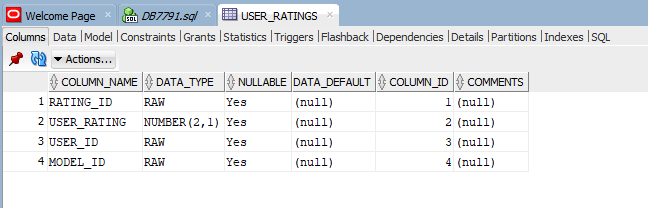
**Models:** This entity stores the model’s name of the product which is identified by company name and device category.



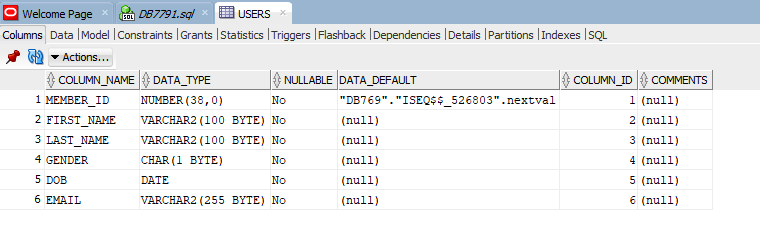
**Items:** It has the details of the item like order ID , name etc.



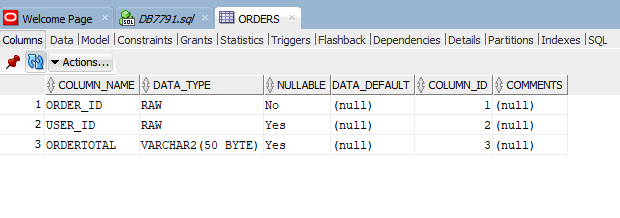
**User\_Ratings:** It stores the user ratings of the products.



**Users:** This entity has the user related information like username, post address, user email and mobile number etc.



**Orders:** This entity stores the user history.



# Data Synthesis

The data for the project has been synthesized using a combination of an online tool named Mockaroo and Microsoft Excel. Some of the prominent functions that were used in Excel include,

* VLOOKUP
* INDEX
* ROWS
* RAND and
* RANDBETWEEN

The tabulation below provides a summary of the data housed in the tables.

|  |  |  |  |
| --- | --- | --- | --- |
| **Table Name** | **Columns** | **# of constraints** | **# of Records** |
| Device\_categories | 2 | 1 | 7 |
| companies | 2 | 2 | 6 |
| models | 3 | 3 | 54 |
| items | 4 | 4 | 11821 |
| orders | 3 | 2 | 9997 |
| Users | 6 | 1 | 2992 |
| Users\_ratings | 4 | 2 | 9508 |
| Category companies | 3 | 3 | 31 |
| Configurations | 10 | 3 | 467 |

# Data Integrity

Data Integrity refers to the consistency and maintenance of the data through the life cycle of the database. In a database, data integrity can be ensured through the implementation of Integrity Constraints in a table. Integrity constraints help apply business rules to the database tables. The constraints can either be at a column level or a table level. Some of the most common constraints are:

* NOT NULL – Prevents a column from having a NULL value.
* PRIMARY KEY – Uniquely identifies each row or record in table.
* FOREIGN KEY – Uniquely identifies a column that references a PRIMARY KEY in another table.
* UNIQUE – Prevents a column from having duplicate values.
* CHECK – Checks for values that satisfy a specific condition as defined by the user.

Listed below are the constraints that were created for our database development project along with their purpose

1. Model\_id: foreign key used to reference the model details in user\_ratings table and configurations table

create table configurations(configuration\_id raw(32),

RAM varchar(100),

diskstorage varchar(100),

weight varchar(100),

batterylife varchar(100),

price varchar(100),

display varchar(100),

os varchar(100),

warranty varchar(100),  
 model\_id raw(32),

constraint configuration\_id\_pk primary key(configuration\_id),

foreign key(model\_id) references models(model\_id) );

create table user\_ratings(rating\_id raw(32),

user\_rating number(2,1),

user\_id raw(32),

model\_id raw(32),

foreign key(user\_id) references userstable(user\_id),

foreign key(model\_id) references models(model\_id) );

1. Unique Constraint: Few columns of tables have Unique constraint as they should be distinct as they should be unique has been included in table definition.

company\_unique ,serial number

1. Not Null Constraint: This has been implemented on different columns as those column values should not be null.

RAM , model\_name

**Queries**

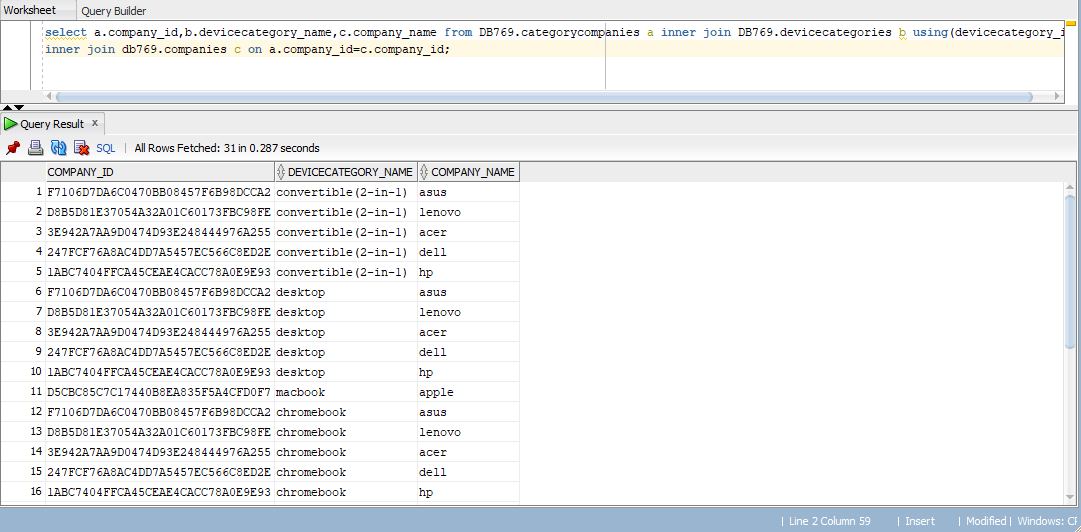
1.Below query provides the details of the company id , company, and specific device names.

select a.company\_id,b.devicecategory\_name,c.company\_name

from DB769.categorycompanies a inner join DB769.devicecategories b

using(devicecategory\_id)

inner join db769.companies c on a.company\_id=c.company\_id;

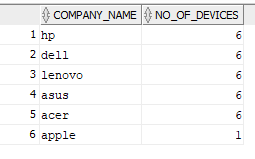
****

2. No of devices respect to the company query:

select c.company\_name,count(c.company\_name)as No\_of\_devices

from DB769.categorycompanies a

inner join db769.companies c on a.company\_id=c.company\_id group by company\_name ;

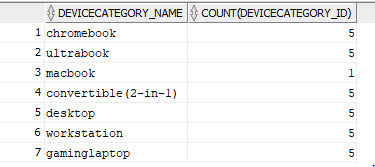


3.No of devices with respect to device category:

select devicecategory\_name,count(devicecategory\_id) from DB769.categorycompanies a

inner join DB769.devicecategories b

using(devicecategory\_id)group by devicecategory\_id,devicecategory\_name;



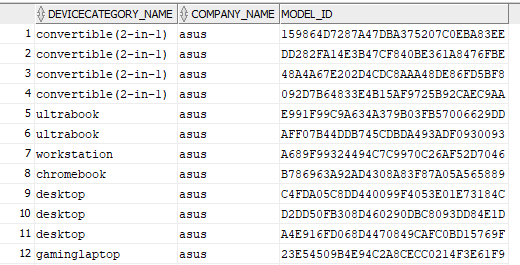
4.To get device category name and company name with providing model id in below query.

select b.devicecategory\_name,c.company\_name,d.model\_id from DB769.categorycompanies a

inner join DB769.devicecategories b using(devicecategory\_id)

inner join db769.companies c on a.company\_id=c.company\_id

inner join DB769.models d on a.categorycompany\_id=d.CATEGORYCOMPANY\_ID;



5.Below query provides details according to names of company ,device and model, price having best ratings .

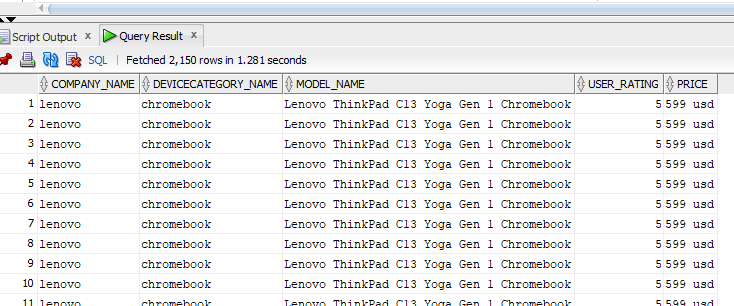
select e.company\_name,f.devicecategory\_name,a.model\_name,c.user\_rating,b.price

from DB769.models a inner join DB769.configurations b using(model\_id)

inner join db769.user\_ratings c using (model\_id) inner join db769.categorycompanies d

using(categorycompany\_id)inner join db769.companies e using (company\_id)inner join db769.devicecategories f using (devicecategory\_id)

where c.user\_rating=5;



6. Count and names of the companies with 5 star device rating:

select count(\*) ,f.devicecategory\_name,e.company\_name from DB769.models a

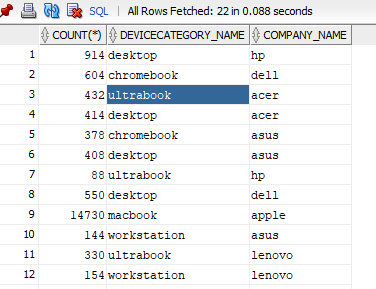
inner join DB769.configurations b using(model\_id) inner join db769.user\_ratings c using (model\_id)

inner join db769.categorycompanies d using(categorycompany\_id)

inner join db769.companies e using (company\_id)

inner join db769.devicecategories f using (devicecategory\_id)

where c.user\_rating=5 group by f.devicecategory\_name,e.company\_name;



7. Stored procedures used:

7.1. Stored proc to insert new records into category company table.

Graphical user interface, text, email

Description automatically generated

This stored proc takes company\_id and devicecategory\_id as input parameters

7.2 Stored procedure to update rating for a model :

Graphical user interface, text, application, email

Description automatically generated

This stored proc takes model\_id as input parameter , calculates avg rating and updates it into models table for the given model\_id

7.3 Stored procedure to update order\_total in orders table

Graphical user interface

Description automatically generated with medium confidence

This stored proc takes order\_id as input parameter and calculates order\_total based on the prices for the items related to order.

**Performance Tuning**

**INDEX:**

Indexing is used to optimize the performance of a database by minimizing the number of disk accesses required when a query is processed.

In this we have three cases:

**Case 1:**

Considering a use case where a particular user\_rating needs to be retrieved based on user\_id.If we fetch the information without indexing, the Cost of CPU is 23

After we create an index on user\_id then Cost of CPU decreases to 5 as shown in the below image.

1 A. Before Indexing

Text, table

Description automatically generated

1 B. After applying indexing

Text, table

Description automatically generated with medium confidence

**CASE 2:** Consider a case where a user wants to know about which model has good rating .For example get all models with average rating > 3 etc. For such scenarios if we execute a query without indexing the cost of CPU is 24.

After creating a combined index on both model\_id,user\_rating the cost of CPU decreases to 14.

1. Before Indexing:

Text

Description automatically generated

1. After Applying aggregate indexingTable

   Description automatically generated

In this case after applying the aggregate indexing the cost drops to 14.

**Case 3:**

Considering the case where user wants to know about number of specific ratings about a given model. Without any indexing, the cost of CPU is 23.But after creating the combined index, the Cost reduces to 2.

Before Indexing :

Graphical user interface, text, application, email

Description automatically generatedINDEXING Range 1 Before

CPU cost in this case is 23

B. After applying indexing to the query :

Text

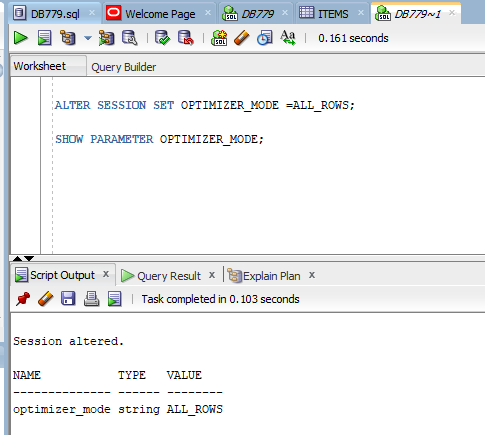
Description automatically generated with medium confidence

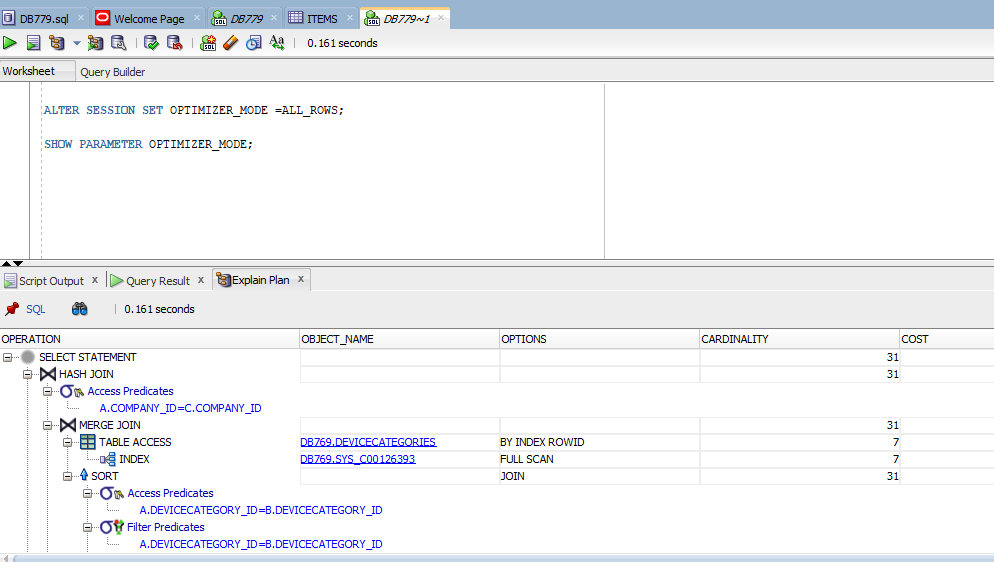
Indexing 1 range after

After applying Indexing the cost of the CPU drops from 23 to 2.

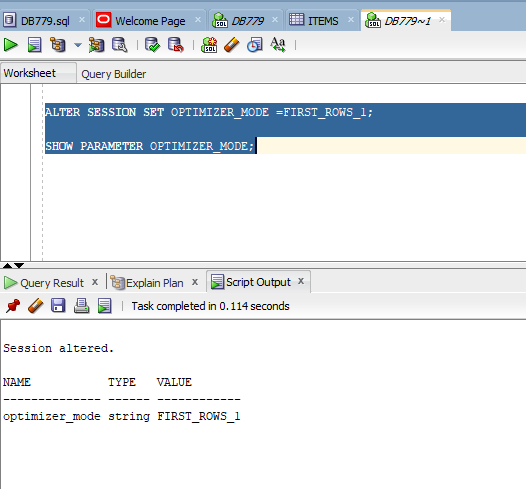
**Optimizer Modes**

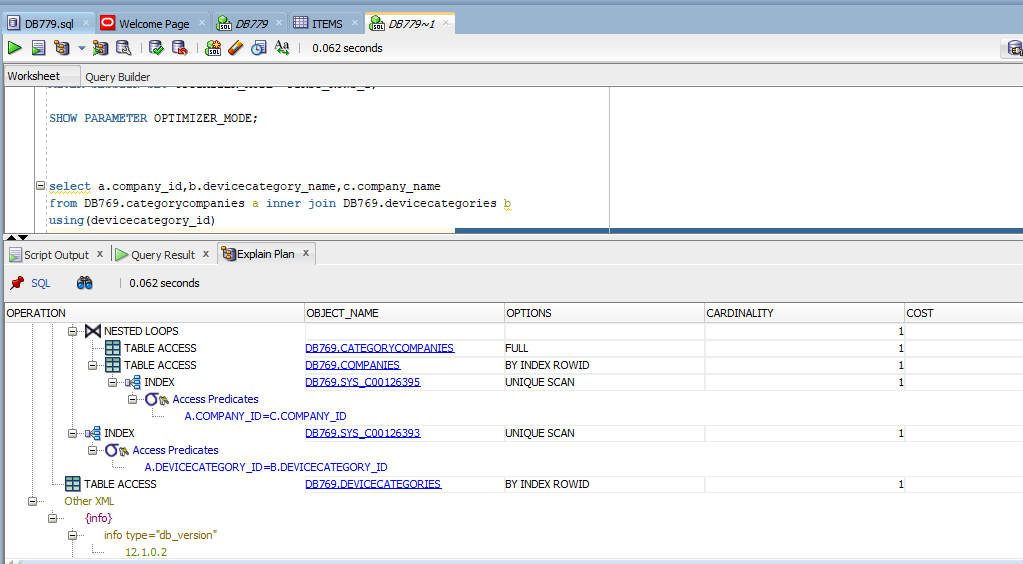
Below example illustrates the use of the OPTIMIZER\_MODE in the Oracle DBMS. We ask the optimizer to run the entire query as efficiently as possible (ALL\_ROWS) or get some small number of rows as quickly as possible (FIRST\_ROWS\_n).

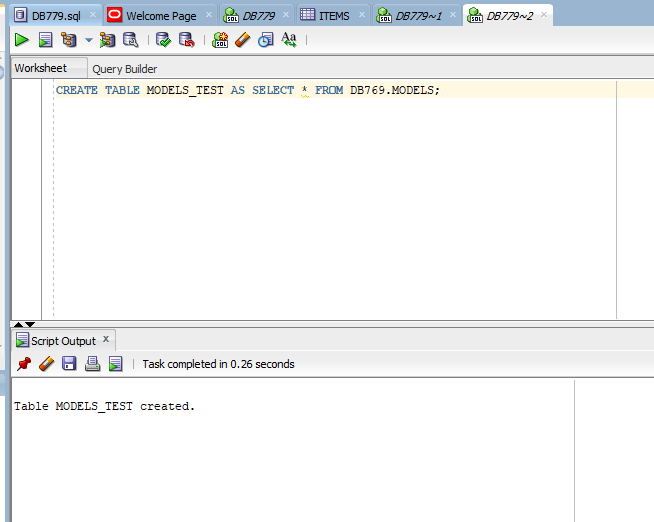




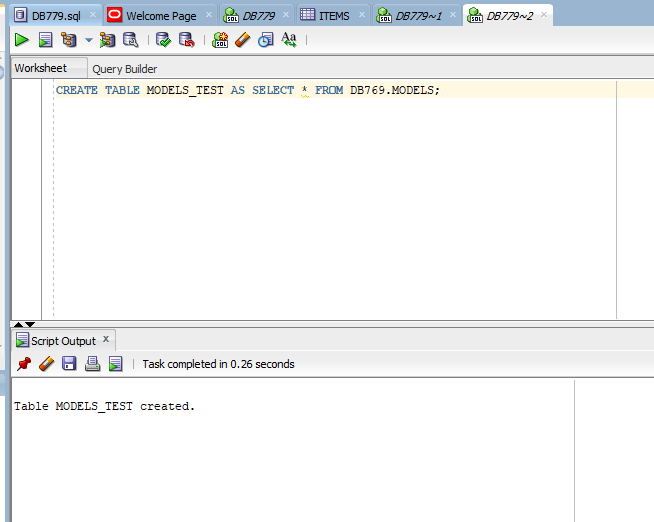
**Step 2**: Next, set the optimizer to FIRST\_ROWS\_1 (asking for the first row as fast as possible). So, now the optimizer will try to develop an execution plan that delivers the first row as efficiently as possible

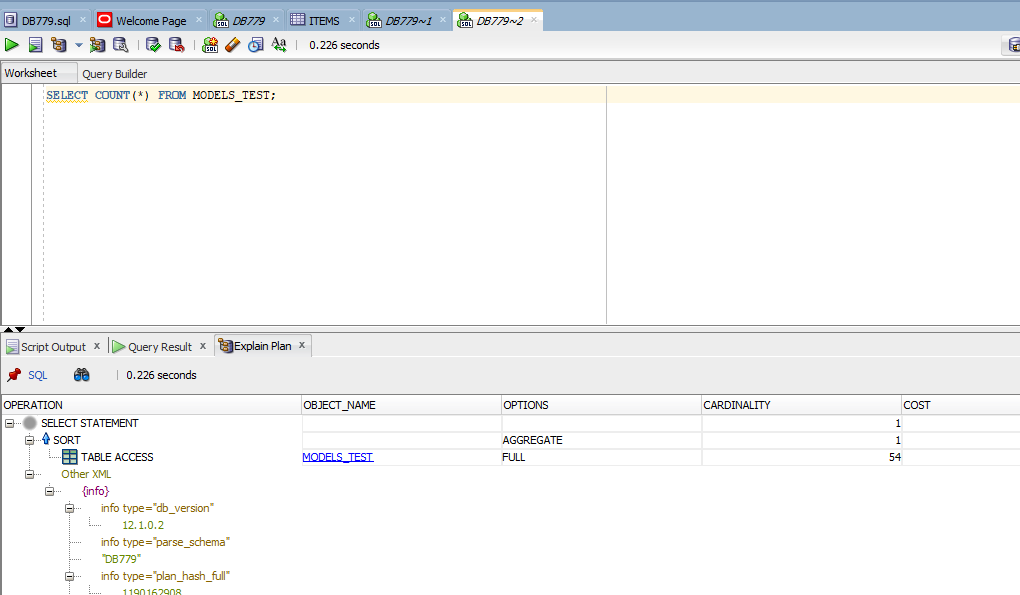




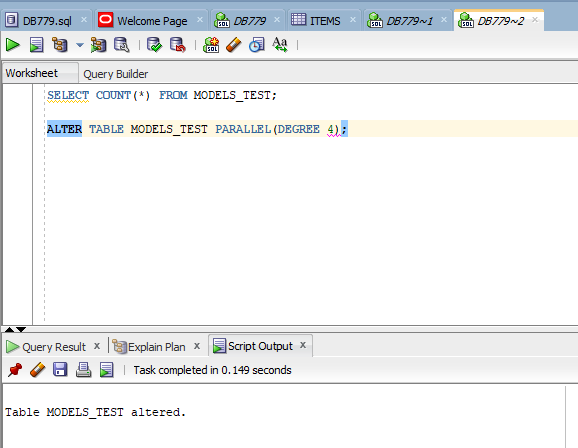


**Basic Parallel Execution:** Parallelism is a natural fit in relational database environments. Large-scale queries, index creation, and other expensive operations often benefit from parallel execution. It is very easy to implement parallel execution since all we really need to do is signal the optimizer. An alter table statement can be used to set the DEGREE of parallelism. Below example illustrates the basics of parallel execution in an Oracle DBMS. This example enables parallel execution for a simple "row count" query.

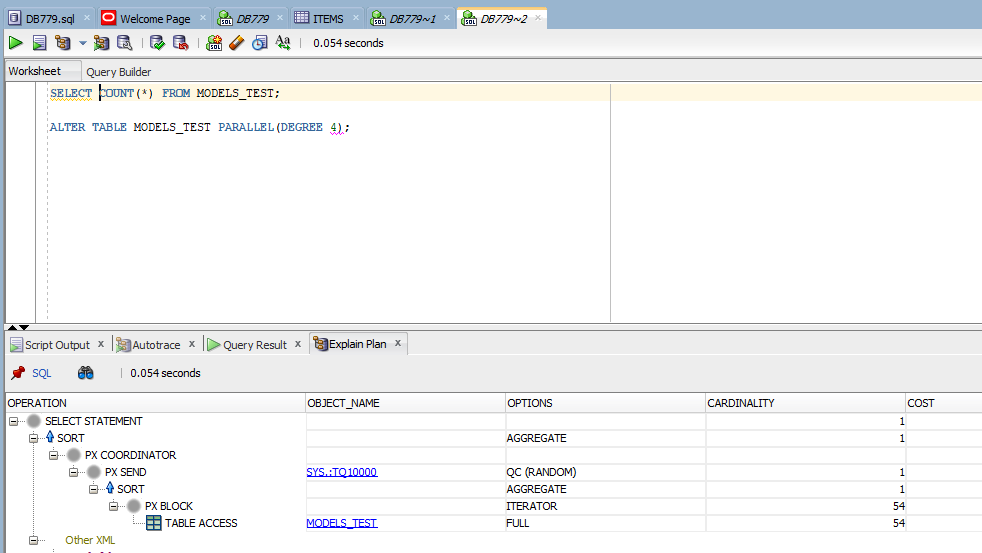




Next, we alter the table to use parallelism. In Oracle, we can specify the DEGREE of parallelism (the number of query processes).

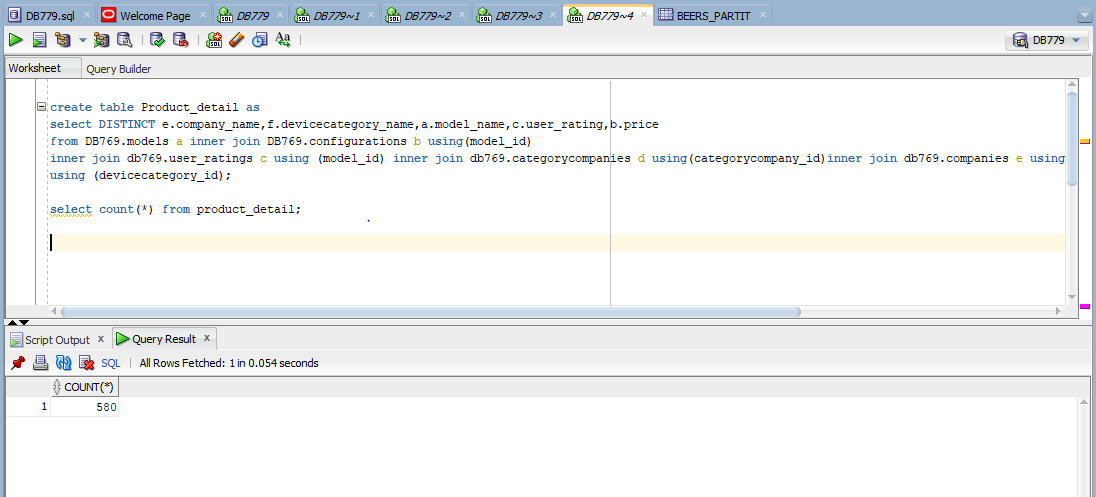


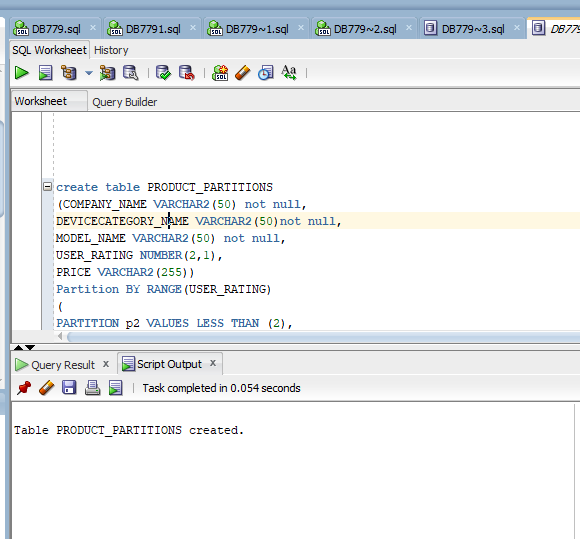
Now, when we run the query again, we see the new execution plan which includes a PX (parallel execution) coordinator and other PX steps as follows:



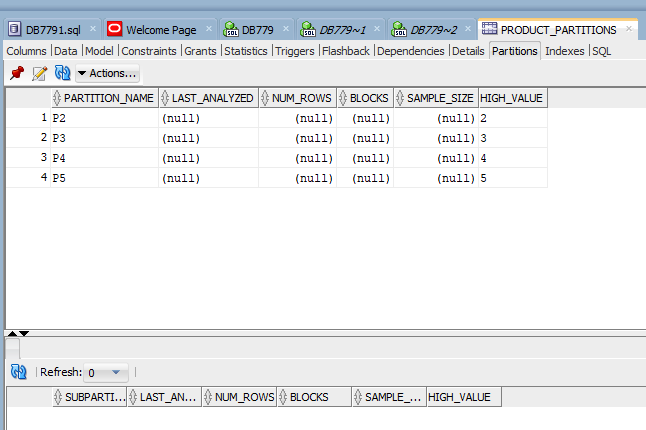
**PARTITIONING:**

Product details table contains all the details of the product mainly user ratings which used as partition. Contains 508 distinct rows .

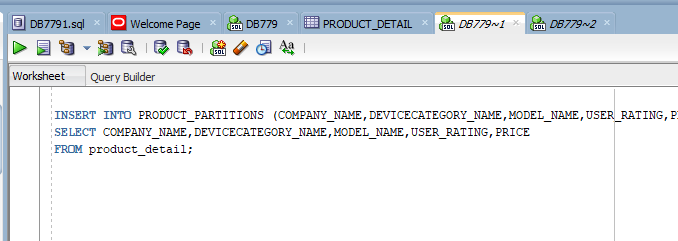


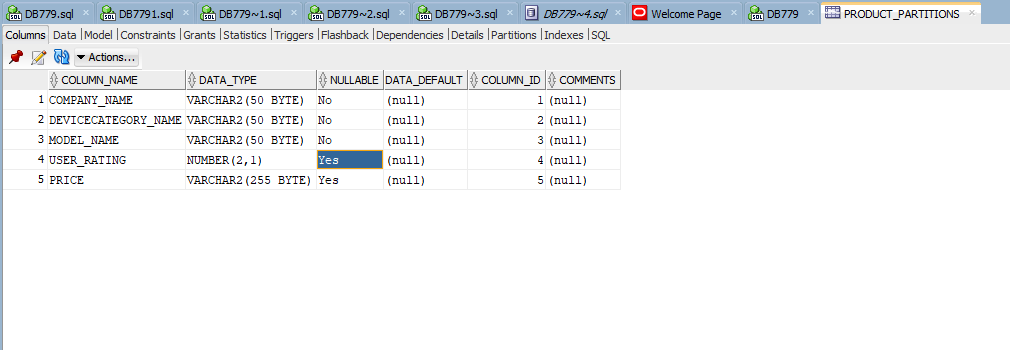


**Product \_partition table constraints and partition before data insertion:**



**Inserting product details into product partition query based on user\_rating**





Product partition query segregates the product detail based on the user rating.

**DATA Visualization using R**

Here,we have visualized the sales by company in R using the results obtained from a sql query .

Graphical user interface, text, application, email

Description automatically generated

**Image 1**

Above diagram lists results of the sales for each company in SQL. Using these results, a pie chart is constructed for comparison of the sales as shown below.

**Graphical user interface, chart

Description automatically generated with medium confidence**

**Image 2**

From the above Pie chart we can observe that Apple has the highest sales followed by Asus with Acer has least sales.

**Case 2:**

In this case we are using Histogram to showcase the total number of ratings for each category.

For this we have exported the user\_ratings data from database and loaded the same into R studio. By running a loop we have calculated the count for each rating and plotted a histogram.

Below diagram depicts the total count for each respective rating.

Chart, bar chart

Description automatically generated

**Image 3**

**Case 3:**

In this case we have visualized the rating of Apple company in R using the results obtained from a SQL query as shown below.

Graphical user interface, application, Word

Description automatically generated

**Image 4**

. Using these results, a pie chart is constructed for ratings of Apple company as shown below.

Graphical user interface, chart, pie chart

Description automatically generated

From the above Pie-chart we can see that more than 70 percent of the users have given a positive rating [>= 3 stars] thereby we can conclude that the user is satisfied with Apple company MacBook’s.