

ADVANCED DATABASE TECHNIQUES PRACTICAL

Practical No. 1

Create different types that include attributes and methods. Define tables for these types by adding a sufficient number of tuples. Demonstrate insert, update and delete operations on these tables. Execute queries on them.

Solution :

Step 1 : Start → www.oracle.com → Sign in OR Sign up

The screenshot shows the Oracle Cloud console interface. At the top, there's a navigation bar with the URL 'console.us-ashburn-1.oraclecloud.com'. Below the navigation bar is a 'Quick Actions' section with four main categories: COMPUTE (Create a VM instance), AUTONOMOUS TRANSACTION PROCESSING (Create an ATP database), AUTONOMOUS DATA WAREHOUSE (Create an ADW database), and NETWORKING (Set up a network with a wizard). To the right of these is a 'Collapsible' sidebar containing 'Account Center' (User Management, Billing, Analyze costs) and 'What's New' (MySQL in Oracle Cloud, Oracle Cloud Infrastructure Vault service integration, Lifecycle state metric for instances, MySQL Database Service availability). The central area features a 'Start Exploring' section with links to 'Key Concepts and Terminology', 'Get Started with FREE training from Oracle University', 'Introduction to APEX', and 'Introduction to Resource Manager'. The bottom of the page includes standard footer links like 'Terms of Use and Privacy' and 'Cookie Preferences'.

Step 2 : In search box, type Nosql and in serives select Nosql database

This screenshot shows the Oracle Cloud console after performing a search for 'nosql'. The search results are displayed in the 'Services' section of the main content area. Under the 'NoSQL Database' category, several items are listed: 'Details for NoSQL Database Cloud', 'Using Oracle NoSQL Database Cloud Ser...', 'Using Oracle NoSQL Database Cloud Ser...', 'Using Oracle NoSQL Database Cloud Ser...', and 'Using Oracle NoSQL Database Cloud Ser...'. To the right of the search results, there's a 'Collapsible' sidebar with 'Account Center' and 'What's New' sections, similar to the first screenshot. The bottom of the page includes the same footer links as the first screenshot.

Step 3 : Click on Create Table.

The screenshot shows the Oracle Cloud interface for creating a new table. The URL is <https://console.us-ashburn-1.oraclecloud.com/nosql/tables>. The page title is "Tables in GoLook-OCI-Tutorials Compartment". On the left, there's a sidebar with "List Scope" (set to "COMPARTMENT" with "GoLook-OCI-Tutorials" selected) and "Filters" (set to "Any state"). The main area has a table header with columns: Status, Name, Read capacity (ReadUnits), Write capacity (WriteUnits), Disk storage (GB), and Date created. A modal dialog titled "Create table" is open, showing the "SIMPLE INPUT" mode selected. The "NAME" field contains "Citizens". Below it, the "Primary key columns" section shows one column named "CitizenID" with type "Number". The "Columns" section is currently empty. At the bottom, there are sections for "Reserved Capacity" with input fields for "READ CAPACITY (READUNITS)", "WRITE CAPACITY (WRITEUNITS)", and "DISK STORAGE (GB)". The status bar at the bottom right indicates "Showing 0 items" and "Page 1".

Step 4 : Mention Name of Table and create columns as required.

Type name of columns and their data types in respective box.

This screenshot shows the "Create table" dialog after adding a column. The "NAME" field now contains "Citizens". In the "Primary key columns" section, there is one column named "CitizenID" with type "Number". A checkbox labeled "SET AS SHARD KEY" is present but unchecked. Below the primary key section, there is a "Columns" section which is currently empty. The "Reserved Capacity" section remains the same as in the previous step. The status bar at the bottom right indicates "Showing 1 item" and "Page 1".

ORACLE Cloud nosql

NoSQL Database

Tables in compartment

Create table

Status: DELETED

NAME: Citizens

Primary key columns

ORDER	COLUMN NAME	TYPE
1	CitizenID	Number

SET AS SHARD KEY

+ Another primary key column

Columns

+ Another column

Reserved Capacity

READ CAPACITY (READUNITS)	WRITE CAPACITY (WRITEUNITS)	DISK STORAGE (GB)
Range: 1 to 40,000	Range: 1 to 20,000	Range: 1 to 5,000

Show advanced options

Create table

Date created: Wed, Oct 14, 2020, 5:02:42 PM UTC

Showing 1 item < Page 1 >

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

ORACLE Cloud nosql

NoSQL Database

Tables in compartment

Create table

Status: DELETED

NAME: Person

Primary key columns

COLUMN NAME	TYPE
Name	String

SET AS SHARD KEY

+ Another primary key column

Columns

+ Another column

Reserved Capacity

READ CAPACITY (READUNITS)	WRITE CAPACITY (WRITEUNITS)	DISK STORAGE (GB)
Range: 1 to 40,000	Range: 1 to 20,000	Range: 1 to 5,000

Show advanced options

Create table

Date created: Wed, Oct 14, 2020, 5:02:42 PM UTC

Showing 1 item < Page 1 >

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

ORACLE Cloud nosql

NoSQL Database

Tables in compartment

Create table

Status: DELETED

NAME: Person

Primary key columns

COLUMN NAME	TYPE
Name	String

DEFAULT VALUE: OPTIONAL VALUE IS NOT NULL

Columns

+ Another column

Primary key columns

COLUMN NAME	TYPE
Surname	String

DEFAULT VALUE: OPTIONAL VALUE IS NOT NULL

Reserved Capacity

READ CAPACITY (READUNITS)	WRITE CAPACITY (WRITEUNITS)	DISK STORAGE (GB)
Range: 1 to 40,000	Range: 1 to 20,000	Range: 1 to 5,000

Show advanced options

Create table

Date created: Wed, Oct 14, 2020, 5:02:42 PM UTC

Showing 1 item < Page 1 >

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

The screenshot shows the Oracle Cloud NoSQL Database interface. A modal window titled 'Create table' is open. It contains two columns defined: 'Surname' (String type) and 'Age' (Number type). Both columns have the 'Value is not null' constraint selected. In the 'Reserved Capacity' section, the 'Read capacity (ReadUnits)', 'Write capacity (WriteUnits)', and 'Disk storage (GB)' are all set to 1. The 'Create table' button is at the bottom of the modal.

This screenshot is nearly identical to the previous one, showing the 'Create table' dialog with two columns and reserved capacity settings. The difference is that the 'Create table' button has been clicked, as indicated by the mouse cursor over it.

Table is created.

The screenshot shows the 'Tables in GoLook-OCI-Tutorials Compartment' list view. A success message 'The table Citizens will be created' is displayed above the table. The table lists two entries: 'Citizens' (Status: CREATING) and 'hello_world' (Status: DELETED). The 'Citizens' row has a green checkmark icon next to it. The table includes columns for Status, Name, Read capacity (ReadUnits), Write capacity (WriteUnits), Disk storage (GB), and Date created.

Status	Name	Read capacity (ReadUnits)	Write capacity (WriteUnits)	Disk storage (GB)	Date created
CREATING	Citizens	0	0	0	Wed, Oct 14, 2020, 5:15:55 PM UTC
DELETED	hello_world	0	0	0	Wed, Oct 14, 2020, 5:02:42 PM UTC

console.us-ashburn-1.oraclecloud.com/nosql/tables

ORACLE Cloud nosql US East (Ashburn) The table Citizens will be created

NoSQL Database

Tables in GoLook-OCI-Tutorials Compartment

Create table

Status	Name	Read capacity (ReadUnits)	Write capacity (WriteUnits)	Disk storage (GB)	Date created
ACTIVE	Citizens	1	1	1	Wed, Oct 14, 2020, 5:16:56 PM UTC
DELETED	hello_world	0	0	0	Wed, Oct 14, 2020, 5:02:42 PM UTC

Showing 2 items < Page 1 >

Terms of Use and Privacy Cookie Preferences Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Step 5 : Click on Table Name.

console.us-ashburn-1.oraclecloud.com/nosql/tables

ORACLE Cloud nosql US East (Ashburn) The table Citizens will be created

NoSQL Database

Tables in GoLook-OCI-Tutorials Compartment

Create table

Status	Name	Read capacity (ReadUnits)	Write capacity (WriteUnits)	Disk storage (GB)	Date created
ACTIVE	Citizens	1	1	1	Wed, Oct 14, 2020, 5:16:56 PM UTC
DELETED	hello_world	0	0	0	Wed, Oct 14, 2020, 5:02:42 PM UTC

Showing 2 items < Page 1 >

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

console.us-ashburn-1.oraclecloud.com/nosql/tables/Citizens/_/edit

ORACLE Cloud nosql US East (Ashburn) The table Citizens will be created

Table information Tags

OCID: ...vn75retuna Show Copy
Compartment: GoLook-OCI-Tutorials
Date created: 2020-10-14T17:16:56.478Z
Time to live (Days):
Read capacity (ReadUnits): 1
Write capacity (WriteUnits): 1
Disk storage (GB): 1

Resources

Columns

Add columns

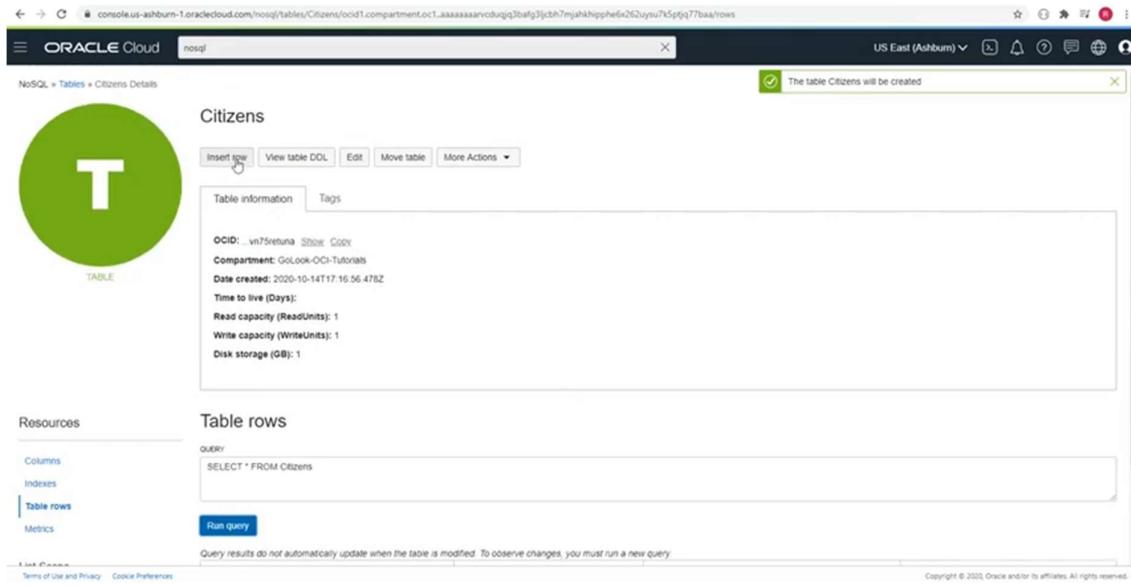
Primary key	Column name	Type	Shard key	Not null
Yes	CitizenID	NUMBER	Yes	Yes
No	Name	STRING	No	No
No	Surname	STRING	No	No
No	Age	NUMBER	No	No

Showing 4 items

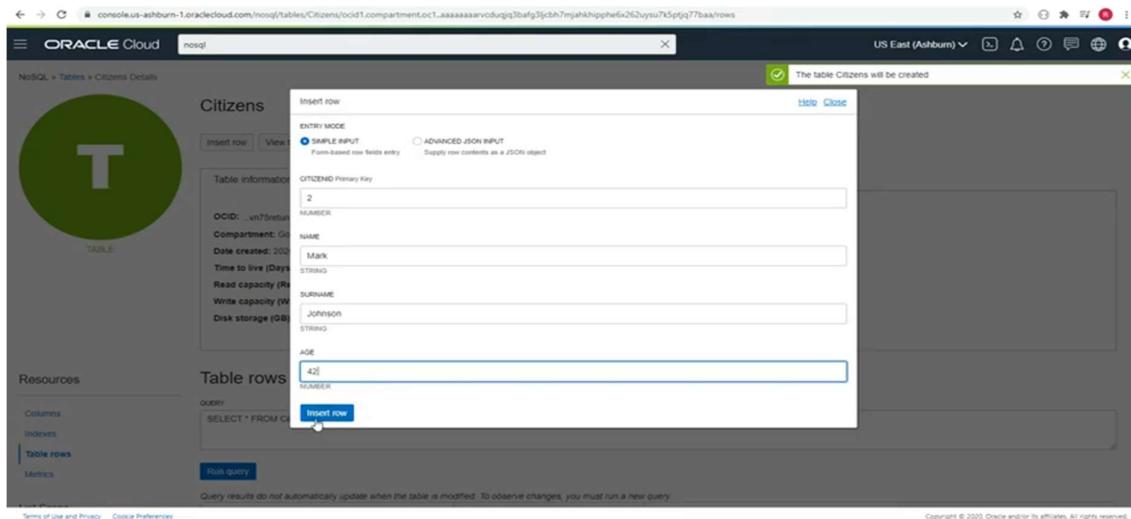
console.us-ashburn-1.oraclecloud.com/nosql/tables/Citizens/_/edit

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Step 6 : To insert rows , click on Insert Row from above option.

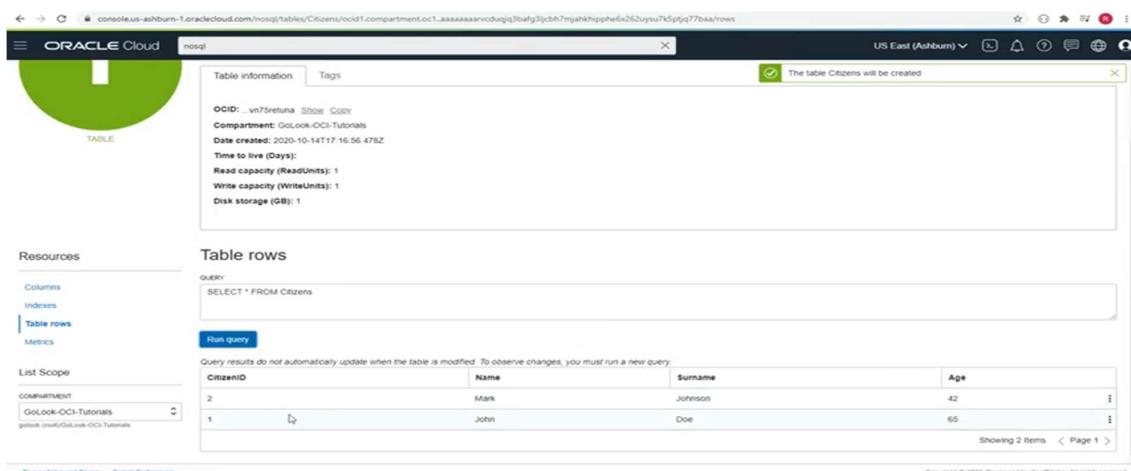


The screenshot shows the Oracle Cloud NoSQL Table Details page for a table named 'Citizens'. At the top, there is a green circular icon with a white letter 'T' and the word 'TABLE' below it. Below the icon, there are tabs for 'Table information' and 'Tags'. A prominent 'Insert row' button is highlighted with a red circle. Other buttons include 'View table DDL', 'Edit', 'Move table', and 'More Actions'. The 'Table information' section displays details such as OCID, Compartment, Date created, Time to live (Days), Read capacity (ReadUnits), Write capacity (WriteUnits), and Disk storage (GB). On the left sidebar, 'Table rows' is selected. In the main content area, there is a 'Table rows' section with a 'QUERY' input field containing 'SELECT * FROM Citizens' and a 'Run query' button. A note at the bottom states: 'Query results do not automatically update when the table is modified. To observe changes, you must run a new query.' A status message at the top right says 'The table Citizens will be created' with a checkmark icon.



This screenshot shows the same Oracle Cloud NoSQL Table Details page as the previous one, but with a modal dialog box titled 'Insert row' overlaid. The dialog has two tabs: 'SIMPLE INPUT' (selected) and 'ADVANCED JSON INPUT'. Under 'SIMPLE INPUT', there are fields for 'CITIZENID Primary Key' (containing '2'), 'NAME' (containing 'Mark'), 'SURNAME' (containing 'Johnson'), and 'AGE' (containing '42'). Below these fields is a 'Help' link and a 'Close' button. The background of the main page is dimmed. The status message 'The table Citizens will be created' remains at the top right.

Step 7 : Run Queries.



The screenshot shows the Oracle Cloud NoSQL Table Details page again. The 'Table rows' section now displays the inserted data. The 'QUERY' field still contains 'SELECT * FROM Citizens'. The results table shows two rows:

CitizenID	Name	Surname	Age
2	Mark	Johnson	42
1	Doe	John	65

A note at the bottom of the results table says: 'Showing 2 items < Page 1 >'. The status message 'The table Citizens will be created' is still present at the top right.

The screenshot shows the Oracle Cloud NoSQL interface. At the top, it says "The table Citizens will be created". Below this, there's a table information panel with details: OCID: ..., Compartment: Golook-OCI-Tutorials, Date created: 2020-10-14T17:16:56.478Z, Time to live (Days): 1, Read capacity (ReadUnits): 1, Write capacity (WriteUnits): 1, and Disk storage (GB): 1. On the left, under "Resources", "Table rows" is selected. In the "Table rows" section, there's a query input field containing "SELECT * FROM Citizens WHERE CitizenID = 2" and a "Run query" button. Below the query is a table with one row:

CitizenID	Name	Surname	Age
2	Mark	Johnson	42

At the bottom, there are links for "Terms of Use and Privacy" and "Cookie Preferences", and a copyright notice: "Copyright © 2020, Oracle and/or its affiliates. All rights reserved."

Step 8 : To Update Table

This screenshot shows the same Oracle Cloud NoSQL interface as the previous one, but with a different table row. The table now has two rows:

CitizenID	Name	Surname	Age
2	Mark	Johnson	42
1	John	Doe	65

A context menu is open over the second row (CitizenID 1), with options "Download JSON", "Update row", and "Delete". The "Update row" option is highlighted with a red circle. At the bottom of the page, there are links for "Terms of Use and Privacy" and "Cookie Preferences", and a copyright notice: "Copyright © 2020, Oracle and/or its affiliates. All rights reserved."

The screenshot shows the Oracle Cloud NoSQL Table Information page for a table named 'Citizens'. A modal window titled 'Update row' is open, showing the entry mode as 'SAMPLE INPUT' (Form-based row fields entry). The primary key 'CITIZENID' is set to 'READ-ONLY'. The row being updated has 'CitizenID' 2, 'Name' 'Steven', 'Surname' 'Johnson', and 'Age' 42. The modal includes a 'Run query' button and an 'Update row' button.

CitizenID	Name	Surname	Age
2	Steven	Johnson	42
1	John	Doe	65

After updating

The screenshot shows the Oracle Cloud NoSQL Table Information page after the update. The table now contains two rows:

CitizenID	Name	Surname	Age
2	Steven	Johnson	42
1	John	Doe	65

Step 9 : For deleting row.

The screenshot shows the Oracle Cloud NoSQL Table Information page with the table rows. The row for CitizenID 1 (Name: John, Surname: Doe, Age: 65) has a context menu open, showing options: 'Download JSON', 'Update row', and 'Delete'.

CitizenID	Name	Surname	Age
2	Steven	Johnson	42
1	John	Doe	65

The screenshot shows the Oracle Cloud NoSQL Table Details page for a table named 'Citizens'. A modal dialog box is open, asking 'Are you sure you want to delete the selected row?' with 'Delete' and 'Cancel' buttons. The main table rows section shows two entries: Steven Johnson (Age 42) and John Doe (Age 65). The 'Table rows' section contains a query input field with 'SELECT * FROM Citizens' and a 'Run query' button.

Step 10 : For deleting Table

The screenshot shows the Oracle Cloud NoSQL Table Details page for the 'Citizens' table. The 'More Actions' dropdown menu is open, indicating the table has been deleted. The table rows section now displays a single entry: Steven Johnson (Age 42). The 'Table rows' section contains a query input field with 'SELECT * FROM Citizens' and a 'Run query' button.

console.us-ashburn-1.oraclecloud.com/nosql/tables/Citizens/ocid1.compartment.oc1..aaaaaaaaarvcduqj3bafg3jcbh7mjahkhpphe6z62uysu7k5ptq77baa/rows

ORACLE Cloud nosql

NoSQL > Tables > Citizens Details

Citizens

TABLE: T

Insert row View table DDL Edit Move table More Actions ▾

Table information Tags

OCID: ...vn75retuna Share Copy
Compartment: GoLook-OCI-Tutorials
Date created: 2020-10-14T17:16:56.478Z
Time to live (Days):
Read capacity (ReadUnits): 1
Write capacity (WriteUnits): 1
Disk storage (GB): 1

Delete Add Tags

Resources Table rows Metrics

Table rows

QUERY: SELECT * FROM Citizens

Run query

Query results do not automatically update when the table is modified. To observe changes, you must run a new query.

Terms of Use and Privacy Cookie Preferences Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

console.us-ashburn-1.oraclecloud.com/nosql/tables/Citizens/ocid1.compartment.oc1..aaaaaaaaarvcduqj3bafg3jcbh7mjahkhpphe6z62uysu7k5ptq77baa/rows

ORACLE Cloud nosql

NoSQL > Tables > Citizens Details

Citizens

TABLE: T

Insert row View table DDL Edit Move table More Actions ▾

Table information Tags

OCID: ...vn75retuna Share Copy
Compartment: GoLook-OCI-Tutorials
Date created: 2020-10-14T17:16:56.478Z
Time to live (Days):
Read capacity (ReadUnits): 1
Write capacity (WriteUnits): 1
Disk storage (GB): 1

Delete

Delete table

Are you sure you want to delete the table named "Citizens"?

Resources Table rows Metrics

Table rows

QUERY: SELECT * FROM Citizens

Run query

Query results do not automatically update when the table is modified. To observe changes, you must run a new query.

Terms of Use and Privacy Cookie Preferences Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

console.us-ashburn-1.oraclecloud.com/nosql/tables/

ORACLE Cloud nosql

NoSQL Database

Tables in GoLook-OCI-Tutorials Compartment

Create table

Status	Name	Read capacity (ReadUnits)	Write capacity (WriteUnits)	Disk storage (GB)	Date created
DELETED	Citizens	0	0	0	Wed, Oct 14, 2020, 5:19:51 PM UTC
DELETED	hello_world	0	0	0	Wed, Oct 14, 2020, 5:02:42 PM UTC

List Scope

COMPARTMENT: GoLook-OCI-Tutorials

Filters

STATE: Any state

Showing 2 items < Page 1 >

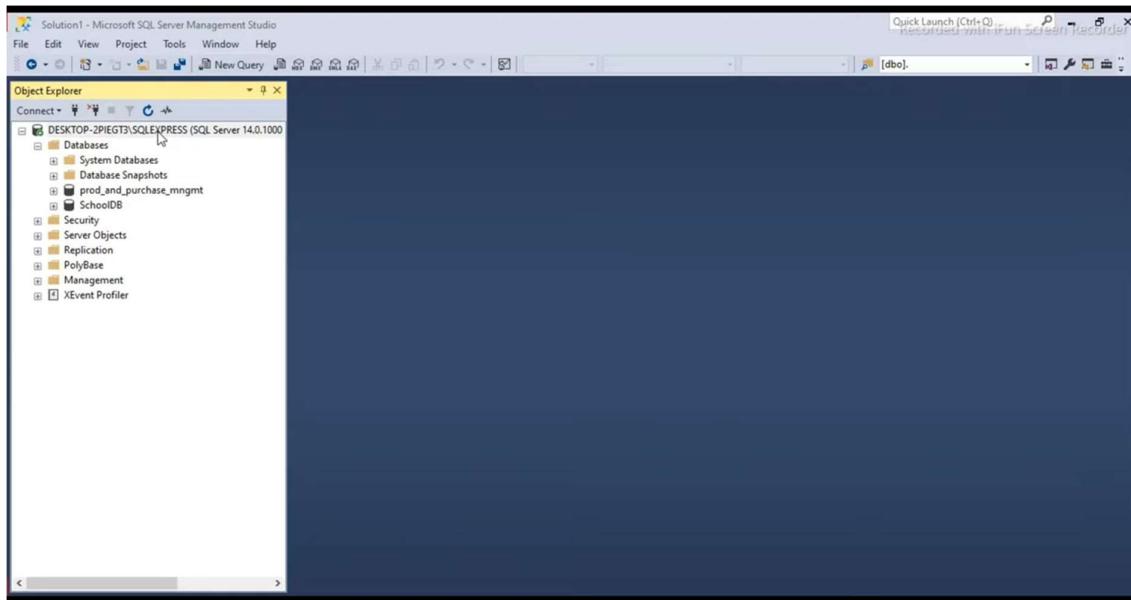
Terms of Use and Privacy Cookie Preferences Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Practical No 2

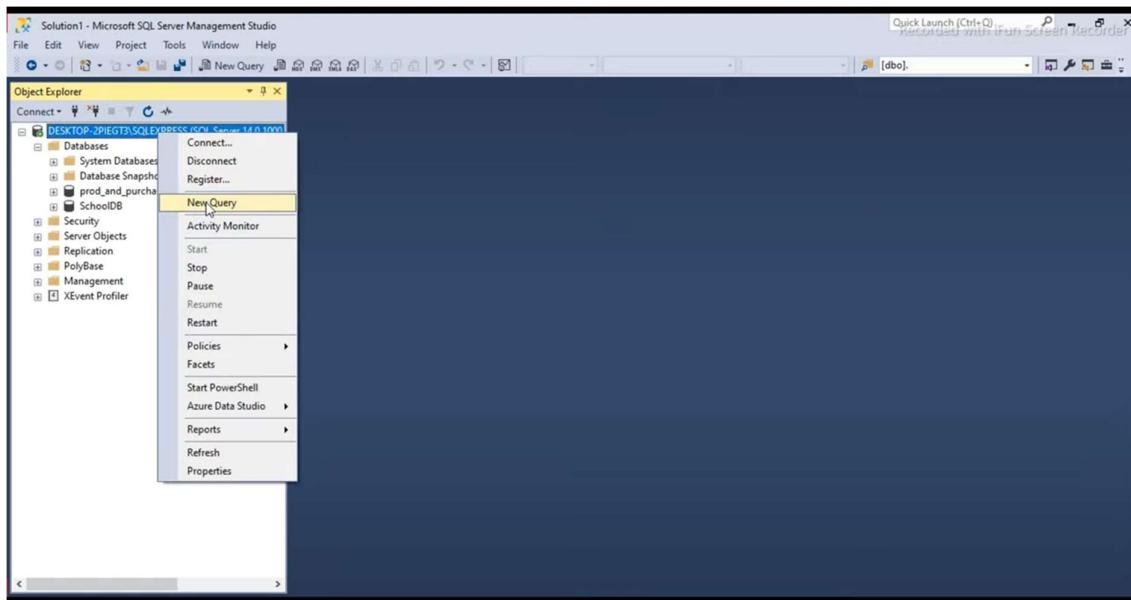
Create an XML database and demonstrate insert, update and delete operations on these tables. Issue queries on it.

Solution :

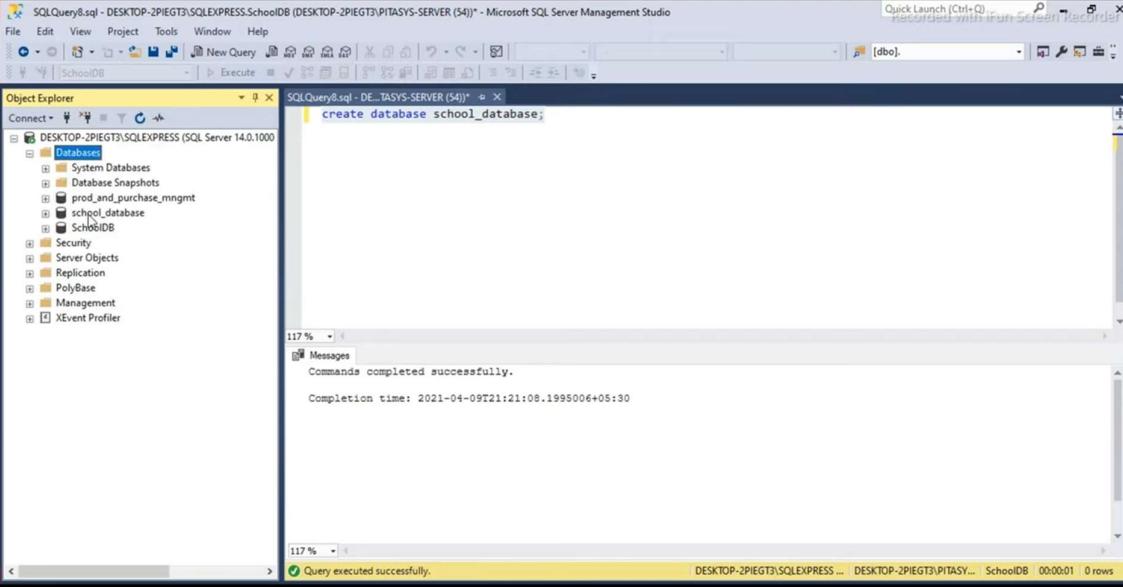
Step 1 : Open Microsoft SQL Server Management Studio.



Step 2 : Right click on Device name and then select New Query.

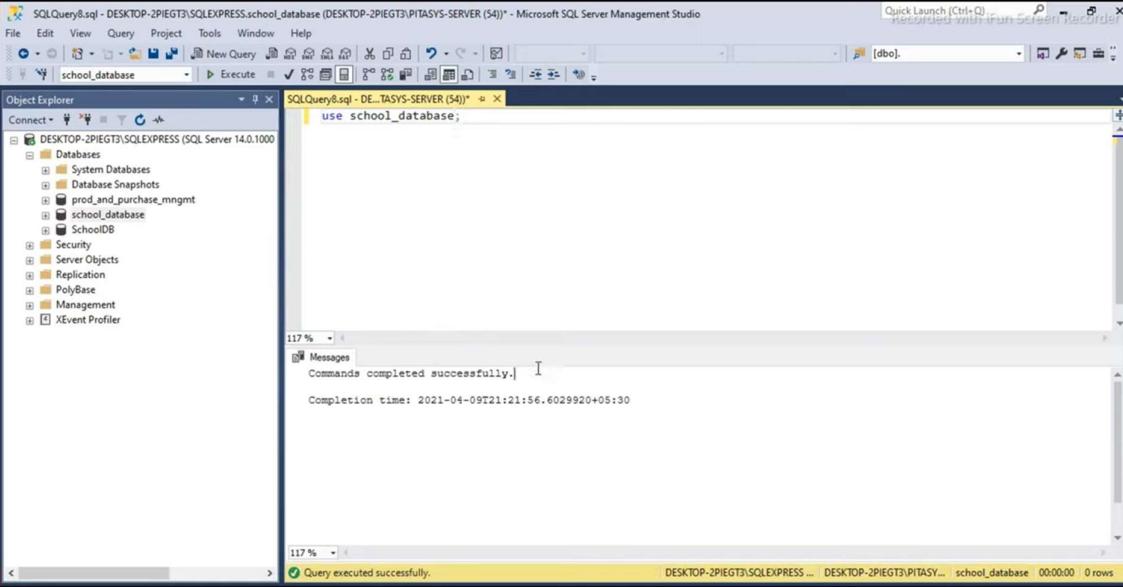


Step 3 : To create database, pass command **create database** and type name of database which you want to create and then execute the query.



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, the connection is to DESKTOP-2PIEGT3\SQLEXPRESS (SQL Server 14.0.1000). A database named 'SchoolDB' is selected. In the center pane, a query window titled 'SQLQuery8.sql - DE...TASY-SERVER (54)*' contains the command: 'create database school_database;'. Below the query window, the 'Messages' pane displays the output: 'Commands completed successfully.' and 'Completion time: 2021-04-09T21:21:08.1995006+05:30'. At the bottom of the screen, a status bar shows 'DESKTOP-2PIEGT3\SQLEXPRESS ... DESKTOP-2PIEGT3\PITASY... SchoolDB 00:00:01 0 rows'.

Step 4 : To create table , first set the database which you want by using **use** command.



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, the connection is to DESKTOP-2PIEGT3\SQLEXPRESS (SQL Server 14.0.1000). A database named 'school_database' is selected. In the center pane, a query window titled 'SQLQuery8.sql - DE...TASY-SERVER (54)*' contains the command: 'use school_database;'. Below the query window, the 'Messages' pane displays the output: 'Commands completed successfully.' and 'Completion time: 2021-04-09T21:21:56.6029920+05:30'. At the bottom of the screen, a status bar shows 'DESKTOP-2PIEGT3\SQLEXPRESS ... DESKTOP-2PIEGT3\PITASY... school_database 00:00:00 0 rows'.

Step 5 : For creating table , pass command **create table** and name of table.

Create columns as per requirement as shown below .

To insert columns write query in round brackets() and execute it.

Eg : **insert into students (column_names)**

values('Lalit');

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, the database 'school_database' is selected. In the center, a query window titled 'SQLQuery8.sql - DESKTOP-2PIEGT3\SQLEXPRESS.school_database (DESKTOP-2PIEGT3\PITASY-SERVER (54))' displays the following SQL code:

```

create table students
(
    roll_no int primary key identity(1,1),
    first_name varchar(10),
    middle_name varchar(10),
    last_name varchar(10),
    creation_time datetime default getdate(),
    updation_time datetime
)

```

The status bar at the bottom indicates 'Query executed successfully.' and 'Completion time: 2021-04-09T21:25:52.0793987+05:30'.

Once you inserted all columns then execute the query , a table is created.

Step 6 : To view the table , execute command **select * from students**

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, the database 'school_database' is selected. In the center, a query window titled 'Results' displays the following table:

roll_no	first_name	middle_name	last_name	creation_time	updation_time
1	Kartik	Ramesh	Pande	2021-04-09 21:29:57.490	NULL
2	Sachin	Ramesh	Rane	2021-04-09 21:30:54.623	NULL
3	Lalt	Gajendra	Kolte	2021-04-09 21:32:28.733	NULL

The status bar at the bottom indicates 'Query executed successfully.' and 'Completion time: 2021-04-09T21:29:57.490+05:30'.

Step 7 : To Update table use **update command**.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, the database 'school_database' is selected. In the center, a query window titled 'Results' displays the following table after running an update command:

roll_no	first_name	middle_name	last_name	creation_time	updation_time
1	Kartik	Ramesh	Pande	2021-04-09 21:29:57.490	NULL
2	Rakesh	Ramesh	Rane	2021-04-09 21:30:54.623	NULL
3	Lalt	Gajendra	Kolte	2021-04-09 21:32:28.733	NULL

The status bar at the bottom indicates 'Query executed successfully.' and 'Completion time: 2021-04-09T21:30:54.623+05:30'.

Step 8 : To delete column use delete command.

```
--insert command
insert into students(first_name,last_name,middle_name)
values('Lalit','Kolte','Gajendra');

--update command
update students set first_name='Sachin',middle_name='Lalit' where roll_no=3;

--delete command
delete from students where roll_no=2;
```

(1 row affected)

Completion time: 2021-04-09T21:41:16.8549350+05:30

```
--create command
create table students(
    roll_no int primary key,
    first_name nvarchar(20),
    middle_name nvarchar(20),
    last_name nvarchar(20),
    creation_time datetime default getdate(),
    updation_time datetime
);

--select command
select * from students
select first_name,last_name from students where roll_no=3;
select * from students where middle_name='Ramesh';

--insert command
insert into students(first_name,last_name,middle_name)
values('Lalit','Kolte','Gajendra');
```

roll_no	first_name	middle_name	last_name	creation_time	updation_time
1	Rakesh	Ramesh	Pande	2021-04-09 21:29:57.490	NULL
2	Sachin	Lalit	Kolte	2021-04-09 21:32:28.733	NULL

Query executed successfully.

Step 9 : For executing queries

For eg :

1. **Select from student where middle_name= 'Ramesh';**

```
select * from students where roll_no=1;
select * from students where middle_name='Ramesh';

insert into students(first_name,last_name,middle_name)
values('Lalit','Kolte','Gajendra');
```

roll_no	first_name	middle_name	last_name	creation_time	updation_time
1	Kartik	Ramesh	Pande	2021-04-09 21:29:57.490	NULL
2	Sachin	Ramesh	Rane	2021-04-09 21:30:54.623	NULL

2. **Select from student where roll no=3;**

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure under 'DESKTOP-2PIEGT3\SQLEXPRESS'. A query window titled 'SQLQuery8.sql - DESKTOP-2PIEGT3\SQLEXPRESS.school_database (DESKTOP-2PIEGT3\PITASYS-SERVER (54))' contains the following SQL code:

```
select * from students where roll_no=3;
select * from students where middle_name='Ramesh';

insert into students(first_name,last_name,middle_name)
values('lalit','Kolte','Gajendra');
```

The results pane shows a single row inserted into the 'students' table:

	roll_no	first_name	middle_name	last_name	creation_time	update_time
1	3	Lalit	Gajendra	Kolte	2021-04-09 21:32:28.733	NULL

At the bottom, a status bar indicates 'Query executed successfully.' and 'DESKTOP-2PIEGT3\SQLEXPRESS ... | DESKTOP-2PIEGT3\PITASYS... school_database 00:00:00 1 rows'.

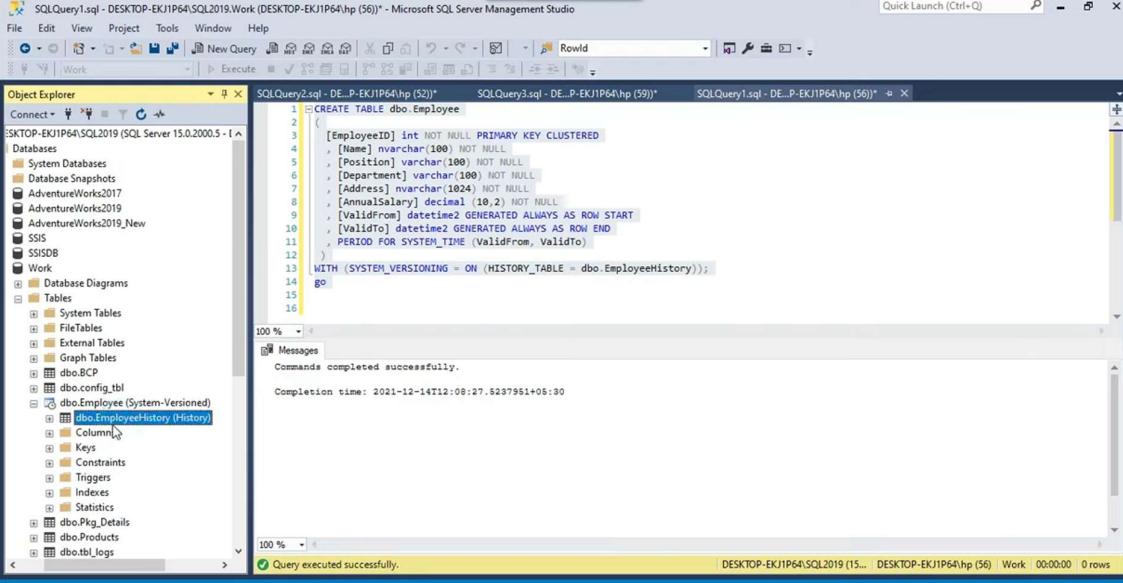
Practical No 3

Create a temporal database and issue queries on it.

Solution :

Step 1 : Start → Open Microsoft SQL Server Management Studio.

Step 2 : Create Temporal table.



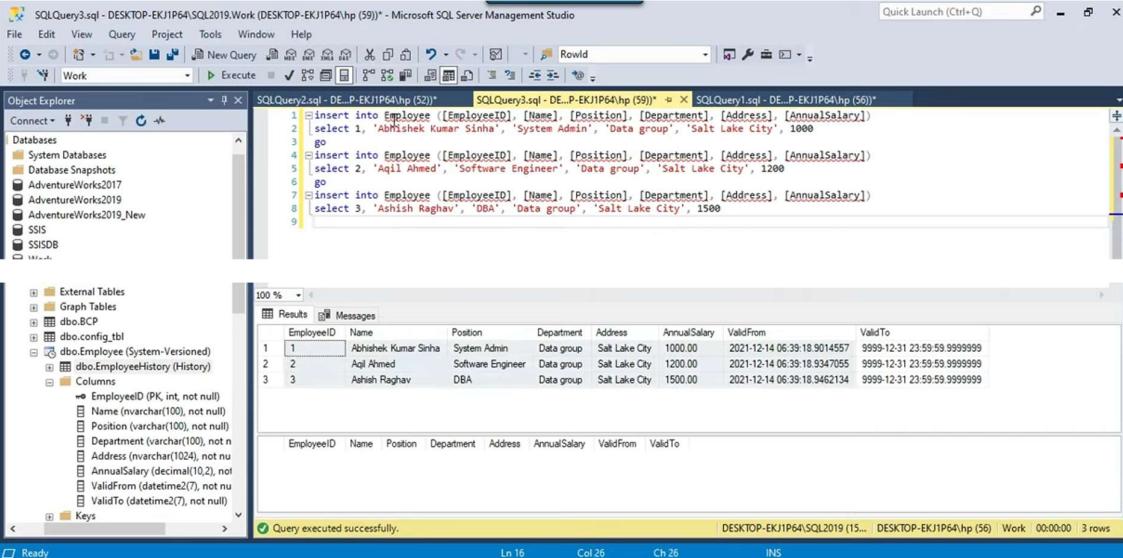
The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, under the 'Work' database, there is a table named 'dbo.EmployeeHistory'. In the center pane, a T-SQL script is being run:

```
CREATE TABLE dbo.Employee
(
    [EmployeeID] int NOT NULL PRIMARY KEY CLUSTERED,
    [Name] nvarchar(100) NOT NULL,
    [Position] varchar(100) NOT NULL,
    [Department] varchar(100) NOT NULL,
    [Address] nvarchar(1024) NOT NULL,
    [AnnualSalary] decimal (10,2) NOT NULL,
    [ValidFrom] datetime2 GENERATED ALWAYS AS ROW START,
    [ValidTo] datetime2 GENERATED ALWAYS AS ROW END,
    PERIOD FOR SYSTEM_TIME (ValidFrom, ValidTo)
)
WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = dbo.EmployeeHistory));
go
```

The status bar at the bottom indicates 'Query executed successfully.' and 'Completion time: 2021-12-14T12:08:27.5237951+05:30'.

System versioned temporal table is created and Employee history table is created as well.

Step 3 : Using **insert** command , insert records in table.



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, under the 'Work' database, there is a table named 'dbo.Employee'. In the center pane, a T-SQL script is being run:

```
1 Insert into Employee ([EmployeeID], [Name], [Position], [Department], [Address], [AnnualSalary])
2 select 1, 'Abhishek Kumar Sinha', 'System Admin', 'Data group', 'Salt Lake City', 1000
3 go
4 Insert into Employee ([EmployeeID], [Name], [Position], [Department], [Address], [AnnualSalary])
5 select 2, 'Aql Ahmed', 'Software Engineer', 'Data group', 'Salt Lake City', 1200
6 go
7 Insert into Employee ([EmployeeID], [Name], [Position], [Department], [Address], [AnnualSalary])
8 select 3, 'Ashish Raghav', 'DBA', 'Data group', 'Salt Lake City', 1500
9
```

In the bottom pane, the results of the query are displayed in a grid:

EmployeeID	Name	Position	Department	Address	AnnualSalary	ValidFrom	ValidTo
1	Abhishek Kumar Sinha	System Admin	Data group	Salt Lake City	1000.00	2021-12-14 06:39:18.9014557	9999-12-31 23:59:59.9999999
2	Aql Ahmed	Software Engineer	Data group	Salt Lake City	1200.00	2021-12-14 06:39:18.9347055	9999-12-31 23:59:59.9999999
3	Ashish Raghav	DBA	Data group	Salt Lake City	1500.00	2021-12-14 06:39:18.9462134	9999-12-31 23:59:59.9999999

The status bar at the bottom indicates 'Query executed successfully.' and 'Completion time: 2021-12-14T12:08:27.5237951+05:30'.

Step 4 : To Update temporal table.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'AdventureWorksLT' is selected. In the center pane, a query window displays the following T-SQL code:

```
SQLQuery2.sql - DESKTOP-EKJ1P64\hp (56) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Object Explorer
Databases
System Databases
Database Snapshots
AdventureWorks2017
AdventureWorks2019
AdventureWorks2019_New
SSIS
SSISDB
Work
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.BCP
dbo.config_tbl
dbo.Employee (System-Versioned)
dbo.EmployeeHistory (History)
Columns
EmployeeID (PK, int, not null)
Name (nvarchar(100), not null)
Position (varchar(100), not null)
Department (varchar(100), not n
Address (nvarchar(1024), not nu
AnnualSalary (decimal(10,2), not
ValidFrom (datetime2(7), not nu
ValidTo (datetime2(7), not null)

16 select * from [dbo].[Employee]
17 select * from [dbo].[EmployeeHistory]
18 go
19 update [Employee]
20 set AnnualSalary=1100
21 where EmployeeID=1
```

The results pane shows two result sets. The first result set displays the current state of the Employee table:

EmployeeID	Name	Position	Department	Address	AnnualSalary	ValidFrom	ValidTo
1	Abhishek Kumar Sinha	System Admin	Data group	Salt Lake City	1100.00	2021-12-14 06:44:30.7312582	9999-12-31 23:59:59.9999999
2	Aql Ahmed	Software Engineer	Data group	Salt Lake City	1200.00	2021-12-14 06:39:18.9347055	9999-12-31 23:59:59.9999999
3	Abhishek Raghav	DBA	Data group	Salt Lake City	1500.00	2021-12-14 06:39:18.9462134	9999-12-31 23:59:59.9999999

The second result set shows the state of the EmployeeHistory table after the update:

EmployeeID	Name	Position	Department	Address	AnnualSalary	ValidFrom	ValidTo
1	Abhishek Kumar Sinha	System Admin	Data group	Salt Lake City	1000.00	2021-12-14 06:39:18.9014557	2021-12-14 06:44:30.7312582

At the bottom, a message indicates "Query executed successfully."

Record is updated.

Step 5 : To delete record from temporal table.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'AdventureWorksLT' is selected. In the center pane, a query window displays the following T-SQL code:

```
SQLQuery2.sql - DESKTOP-EKJ1P64\hp (56) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Object Explorer
Databases
System Databases
Database Snapshots
AdventureWorks2017
AdventureWorks2019
AdventureWorks2019_New
SSIS
SSISDB
Work
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.BCP
dbo.config_tbl
dbo.Employee (System-Versioned)
dbo.EmployeeHistory (History)
Columns
EmployeeID (PK, int, not null)
Name (nvarchar(100), not null)
Position (varchar(100), not null)
Department (varchar(100), not n
Address (nvarchar(1024), not nu
AnnualSalary (decimal(10,2), not
ValidFrom (datetime2(7), not nu
ValidTo (datetime2(7), not null)

16 select * from [dbo].[Employee]
17 select * from [dbo].[EmployeeHistory]
18 go
19 update [Employee]
20 set AnnualSalary=1200
21 where EmployeeID=1
22 go
23 delete from [Employee] where EmployeeID=2
```

The results pane shows two result sets. The first result set displays the current state of the Employee table:

EmployeeID	Name	Position	Department	Address	AnnualSalary	ValidFrom	ValidTo
1	Abhishek Kumar Sinha	System Admin	Data group	Salt Lake City	1200.00	2021-12-14 06:45:56.1656573	9999-12-31 23:59:59.9999999
2	Abhishek Raghav	DBA	Data group	Salt Lake City	1500.00	2021-12-14 06:39:18.9462134	9999-12-31 23:59:59.9999999

The second result set shows the state of the EmployeeHistory table after the delete operation:

EmployeeID	Name	Position	Department	Address	AnnualSalary	ValidFrom	ValidTo
1	Abhishek Kumar Sinha	System Admin	Data group	Salt Lake City	1000.00	2021-12-14 06:39:18.9014557	2021-12-14 06:44:30.7312582
2	Abhishek Kumar Sinha	System Admin	Data group	Salt Lake City	1100.00	2021-12-14 06:44:30.7312582	2021-12-14 06:45:56.1656573
3	Aql Ahmed	Software Engineer	Data group	Salt Lake City	1200.00	2021-12-14 06:39:18.9347055	2021-12-14 06:47:06.3843773

At the bottom, a message indicates "Query executed successfully."

Practical No 4

Demonstrate the use of data management and operations using NoSQL in the Cloud.

Solution :

Step 1 : Start → www.oracle.com → Sign in OR Sign up

Step 2 : Open database management console.

The screenshot shows the Oracle Cloud Database Management interface. The left sidebar has sections for Overview, Fleet Summary, Database Groups (which is selected), and Scope. The main content area is titled "Database Groups in DB-Management-Demo Compartment". It shows a table with one item: "Name: UAT, Number of Databases: 2". A search bar at the top right says "Search by Name". At the bottom, there are links for "Terms of Use and Privacy" and "Cookie Preferences", and a copyright notice "Copyright © 2021, Oracle and/or its affiliates. All rights reserved."

Step 3 : Click on UAT database Group.

The screenshot shows the Oracle Cloud Database Management interface, specifically the "Database Group Details" page for the "UAT" group. On the left, there's a large blue circular icon with "DG" in white. The main content area has tabs for "Pleet Summary", "Edit Description", "Move Resource", and "Delete". Below these are sections for "Database Group Details" and "Managed Databases". The "Managed Databases" section shows a table with two entries: "Finstage" (PDB, compartment: gamnaray1209 (root)/UAT) and "HBUAT" (PDB, compartment: gamnaray1209 (root)/DB-Management-Demo). A search bar at the top right says "Search by Database Display Name". At the bottom, there are links for "Terms of Use and Privacy" and "Cookie Preferences", and a copyright notice "Copyright © 2021, Oracle and/or its affiliates. All rights reserved."

Step 4 : To create Job.

Click on create Job.

The screenshot shows the Oracle Cloud Database Management interface. The top navigation bar includes the Oracle Cloud logo, a search bar, and a dropdown for 'US East (Ashburn)'. Below the navigation is the 'Database Management > Database Groups > Database Group Details' path. The main content area is titled 'UAT' and contains a large blue circle with 'DG' in white. On the left, there's a sidebar with 'Resources' sections for 'Managed Databases', 'Jobs' (which is selected), 'Job Executions', and 'Scope'. The 'Jobs' section has a 'Create Job' button and a table with columns: Job Name, Job Description, Job Status, Created, and Last Updated. A search bar is at the top of the table. The right side of the screen shows 'Database Group Details' with fields for OCID, Compartment ID, and creation date. A 'Time Period' dropdown shows 'Last 60 min' from 'Today 07:45 PM - 08:45 PM UTC'. At the bottom, there are links for 'Terms of Use and Privacy' and 'Cookie Preferences', and a copyright notice: 'Copyright © 2021, Oracle and/or its affiliates. All rights reserved.'

Step 5 : Fill the basic information in the box like Job name , description , SQL type etc.

This screenshot shows the 'Create Job' dialog box overlaid on the Database Group Details page. The dialog is titled 'Create Job' and asks for 'Provide basic information for the Job'. It includes fields for 'Job Name' (set to 'releasecheck1'), 'Job Description' (set to 'check before release'), 'SQL Type' (set to 'Query'), 'Choose Compartment' (set to 'DB-Management-Demo'), and 'Timeout' (set to '1 minute'). Below this, it asks for 'Specify credentials for the connection' with fields for 'Username' (set to 'dbsnmp') and 'Password' (set to '*****'). There's also a 'Role' field set to 'Normal'. At the bottom, there are 'Create Job' and 'Cancel' buttons. The background of the dialog is semi-transparent, allowing the underlying database group details to be seen.

The screenshot shows the Oracle Cloud Database Management interface. On the left, there's a sidebar with 'UAT' and 'DG' icons, and sections for 'Resources' (Managed Databases, Jobs, Job Executions, Scope) and 'Database Group Details' (OCID: rdmaaq, Created: Fri Jan 22, 2021, 8:38:49 PM UTC, Compartment: gannaray1209 (root)/DB-Management-Demo). The main area is titled 'Create Job' and contains fields for 'Specify credentials for the connection' (Username: dbsmmp, Password: *****), 'Oracle Object Storage for Job Results' (Bucket Name: db-management), and an 'SQL Command' section (Load SQL: 1. select pnum,part FROM parts_category;). A 'Create Job' button is highlighted with a blue border.

Job release check has created.

The screenshot shows the Oracle Cloud Database Management interface. On the left, there's a sidebar with 'UAT' and 'DG' icons, and sections for 'Resources' (Managed Databases, Jobs, Job Executions, Scope) and 'Database Group Details' (OCID: rdmaaq, Created: Fri Jan 22, 2021, 8:38:49 PM UTC, Compartment: gannaray1209 (root)/DB-Management-Demo). The main area shows a table of jobs. One job is listed: 'releasecheck1' (Job Name), 'check before release' (Job Description), 'Active' (Job Status), 'Fri, 22 Jan, 2021 20:49:17 UTC' (Created), and 'Fri, 22 Jan, 2021 20:49:17 UTC' (Last Updated). A success message at the top right says 'Job 'releasecheck1' was created successfully.'

Step 6 : To check job execution , click on job name.

The screenshot shows the Oracle Cloud Database Management interface. At the top, there's a navigation bar with 'ORACLE Cloud' and a search bar. Below it, the path 'Database Management > Database Group Details > Job Details' is visible. The main content area displays a job named 'releasecheck1'. On the left, there's a large blue square icon with a white letter 'J' and the word 'INACTIVE' below it. To the right of the icon, there are two buttons: 'Move Job' and 'Delete Job'. A 'Job Details' section contains the following information:

Description: check before release	Compartment: DB-Management-Demo
Job Id: z5wga Show Copy	Schedule Type: Immediate
Job Type: SQL	Job SQL: select pnum,pcat FROM parts_category
Username: dbtemp	Created: Fri, 22 Jan, 2021 20:49:17 UTC
Operation Type: Execute SQL	Last Updated: Fri, 22 Jan, 2021 20:49:46 UTC
SQL Type: Query	

Below this, there are sections for 'Resources' and 'Job Executions'. The 'Job Executions' section has a search bar and a table showing two entries:

Name	Database Name	Status	Duration	Submitted
releasecheck1_HRUAT_2021-01-22T20:49:25.112Z	HRUAT	Succeeded	13.57 seconds	Fri, 22 Jan, 2021 20:49:25 UTC
releasecheck1_Finstage_2021-01-22T20:49:25.112Z	Finstage	Succeeded	13.57 seconds	Fri, 22 Jan, 2021 20:49:25 UTC

At the bottom of the page, a URL is displayed: <https://comedit.us-phoenix-1.oraclecloud.com/dbmgmt-u/jobs/jobExecutions?ocid=oci1:dbmgmtjob.oc1.usamazaaa5k5pqqzphpwfhsddo2j7b42:vene5keedapuz7y...States>All&compareChoices=SHGDV&managedDatabaseGroupIds=oci1:dbmgmtmanageddatabasegroup.oc1.usamazaaa5k5pqqzkmfm7t5sh3kdytagpdych66d132yc19fd-dm0qg>

Step 7 : To check job execution output , click on one of the executions.

This screenshot shows the 'Job Execution Details' page for the 'releasecheck1_HRUAT_2021-01-22T20:49:25.112Z' execution. The top part displays basic job execution details:

Type: SQL	Submitted: Fri, 22 Jan, 2021 20:49:25 UTC
Associated Database: HRUAT	Duration: 13.57 seconds
Associated Job: releasecheck1	Results Bucket: phbnanagegb0/db-management
Username: dbtemp	

Below this, the 'Job Execution Output' section shows the results of the SQL query:

```
1 |   "type": "table",
2 |   "data": [
3 |     {
4 |       "PNUM": 1,
5 |       "PCAT": "ELECTRONICS"
6 |     },
7 |     {
8 |       "PNUM": 2,
9 |       "PCAT": "SPORTS"
10 |    },
11 |    {
12 |      "PNUM": 3,
13 |      "PCAT": "HOME"
14 |    }
15 |  ]
16 |
17 |
```

At the bottom of the page, there are links for 'Terms of Use and Privacy' and 'Cookie Preferences' on the left, and a copyright notice 'Copyright © 2021, Oracle and/or its affiliates. All rights reserved.' on the right.

Step 8 : To delete Job , click on above option Delete Job.

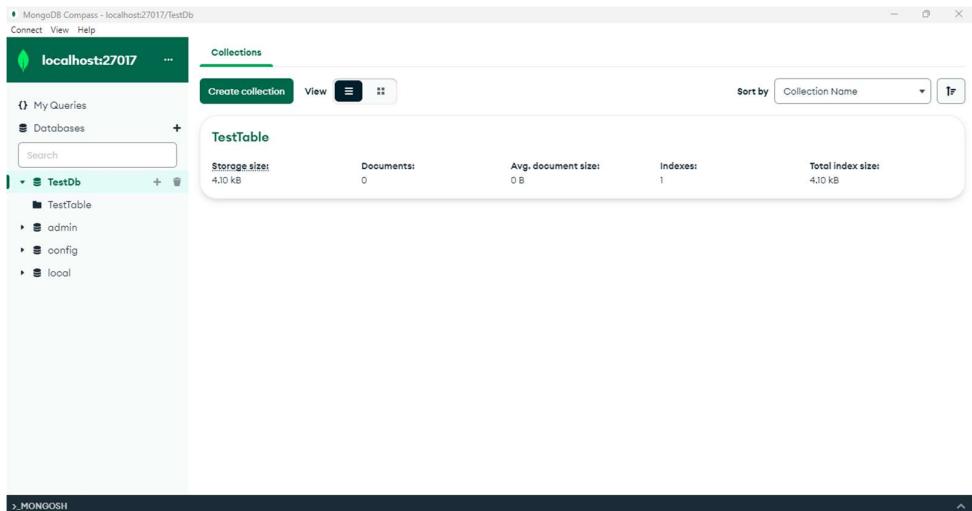
The screenshot shows a web interface for Oracle Cloud Database Management. At the top, there's a navigation bar with the Oracle Cloud logo, a search bar, and account information for 'US East (Ashburn)'. Below the navigation, the path 'Database Management > Database Group Details > Job Details' is visible. A large blue square icon contains a white letter 'J'. To its right, the job name 'releasecheck1' is displayed. Underneath the job name are two buttons: 'Move Job' (gray) and 'Delete Job' (red). A small link 'Job Details' is also present. The main area below the job details is currently empty.

Practical No 5

Demonstrate the Accessing and Storing and performing CRUD operations in MongoDB.

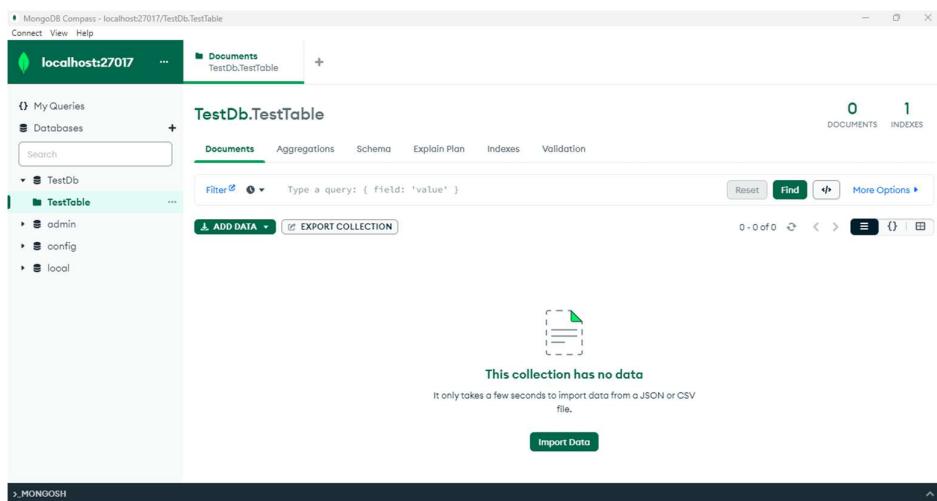
Solution :

Step 1 : Start → Open MongoDB.



The screenshot shows the MongoDB Compass interface. On the left, the sidebar lists databases: My Queries, Databases (TestDb selected), and local. Under TestDb, it shows sub-databases: TestTable, admin, config, and local. The main area displays the 'Collections' tab for the TestTable collection. It shows storage size of 4.10 kB, 0 documents, 0 B avg. document size, 1 index, and 4.10 kB total index size. A 'Sort by' dropdown and a search bar are also present.

Step 2 : Select create database and then click on Add data and then choose Insert document.



The screenshot shows the MongoDB Compass interface for the TestDb.TestTable collection. The sidebar remains the same as the previous screenshot. The main area shows the 'Documents' tab for TestDb.TestTable. It indicates 0 documents and 1 index. A 'Filter' input field with placeholder 'Type a query: { field: 'value' }' is present. Below it are 'ADD DATA' and 'EXPORT COLLECTION' buttons. A message states 'This collection has no data' and provides instructions: 'It only takes a few seconds to import data from a JSON or CSV file.' An 'Import Data' button is at the bottom.

The screenshot shows the MongoDB Compass interface. On the left, the sidebar lists databases (My Queries, Databases) and the current database (TestDb). Within TestDb, the collection 'TestTable' is selected. The main pane displays the 'Documents' tab for 'TestDb.TestTable'. It shows 0 documents and 1 index. A search bar and filter dropdown are at the top. Below them is a button to 'ADD DATA' which includes options for 'Import file' and 'Insert document'. A message states 'This collection has no data' and provides instructions to import from JSON or CSV. A green 'Import Data' button is visible. The bottom status bar shows 'localhost:27017' and 'MONGOSH'.

Insert data like name, age, status etc.

This screenshot shows the 'Insert to Collection TestDb.TestTable' dialog box. The input field contains the following JSON document:

```
1 //+
2 * Paste one or more documents here
3 /*
4 +
5 * "_id": {
6 *   "$oid": "637f8bedbc71a1d4373050bb"
7 * }
```

Below the input field are 'Cancel' and 'Insert' buttons. The background shows the same MongoDB Compass interface as the previous screenshot, with the collection now containing 1 document.

MongoDB Compass - localhost:27017/TestDb.TestTable

localhost:27017

Documents TestDb.TestTable

My Queries Databases

TestDb TestTable

TestDb.TestTable

1 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: 'value' } Reset Find More Options

ADD DATA EXPORT COLLECTION

`_id: ObjectId('637ff8d48bc71a1d4373050bb')`
Name: "Sana"
Age: 25
Status: "Pending"

Step 3 : You can add new data and update it.

MongoDB Compass - localhost:27017/TestDb.TestTable

localhost:27017

Documents TestDb.TestTable

My Queries Databases

TestDb TestTable

TestDb.TestTable

3 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: 'value' } Reset Find More Options

ADD DATA EXPORT COLLECTION

`_id: ObjectId('637ff8d48bc71a1d4373050bb')`
Name: "Sana"
Age: 25
Status: "Pending"

`_id: ObjectId('637ff8db6bc71a1d4373050bc')`
Name: "Payal"
Age: 33
Status: "Completed"

`_id: ObjectId('637ff8deebc71a1d4373050bd')`
Name: "Ramesh"
Age: 28
Status: "Pending"

Step 4 : We can see access data by mentioning its values or names in Filter menu.

MongoDB Compass - localhost:27017/TestDb.TestTable

localhost:27017

Documents TestDb.TestTable

My Queries Databases

TestDb TestTable

TestDb.TestTable

3 DOCUMENTS 1 INDEXES

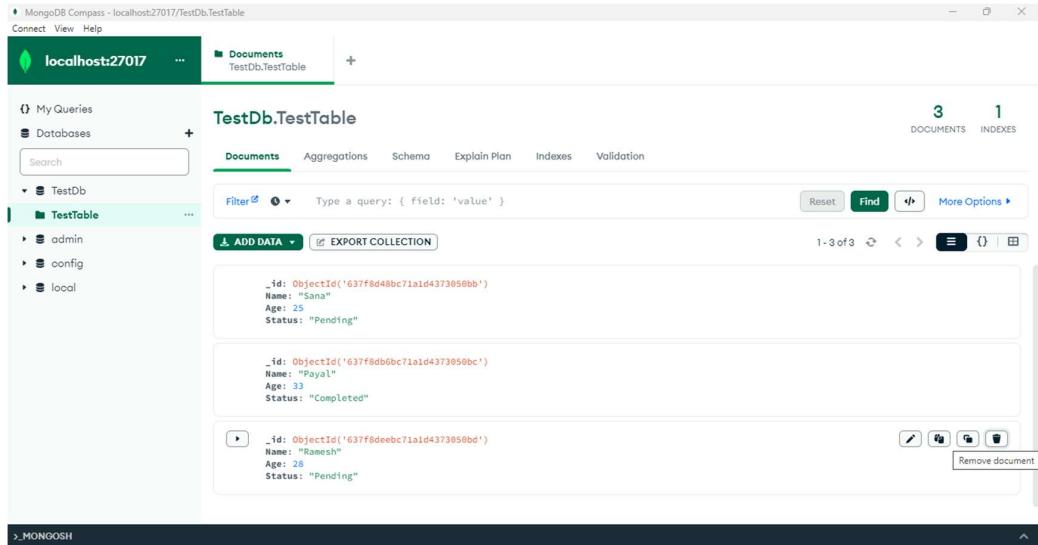
Documents Aggregations Schema Explain Plan Indexes Validation

Filter `{Age:25}` Reset Find More Options

ADD DATA EXPORT COLLECTION

`_id: ObjectId('637ff8d48bc71a1d4373050bb')`
Name: "Sana"
Age: 25
Status: "Pending"

Step 5 : To delete data, click on remove document.

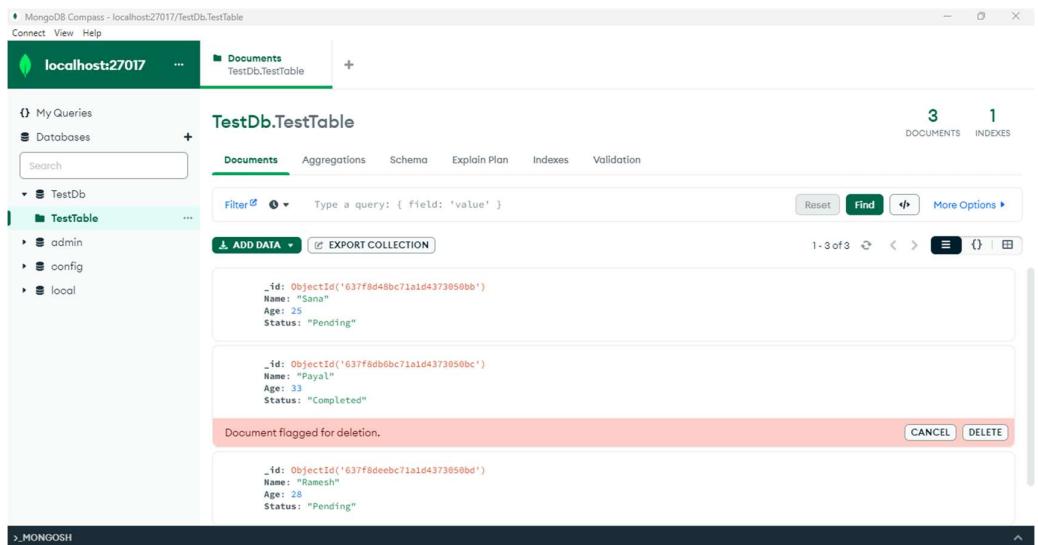


The screenshot shows the MongoDB Compass interface. On the left, the sidebar lists databases (admin, config, local) and collections (TestDb, TestTable). The main area displays the 'Documents' tab for 'TestDb.TestTable'. It shows three documents with the following data:

- `_id: ObjectId('637f8d48bc71a1d4373050bb')`
Name: "Sana"
Age: 25
Status: "Pending"
- `_id: ObjectId('637f8db6bc71a1d4373050bc')`
Name: "Payal"
Age: 33
Status: "Completed"
- `_id: ObjectId('637f8deebc71a1d4373050bd')`
Name: "Ramesh"
Age: 28
Status: "Pending"

On the right, there are buttons for 'Reset', 'Find', 'More Options', and a 'Remove document' button next to the third document's row.

The updated database will look like :



The screenshot shows the same MongoDB Compass interface after a document has been deleted. The 'Documents' tab for 'TestDb.TestTable' now displays only two documents:

- `_id: ObjectId('637f8d48bc71a1d4373050bb')`
Name: "Sana"
Age: 25
Status: "Pending"
- `_id: ObjectId('637f8db6bc71a1d4373050bc')`
Name: "Payal"
Age: 33
Status: "Completed"

A red banner at the bottom indicates 'Document flagged for deletion.' with 'CANCEL' and 'DELETE' buttons.

Practical No 6

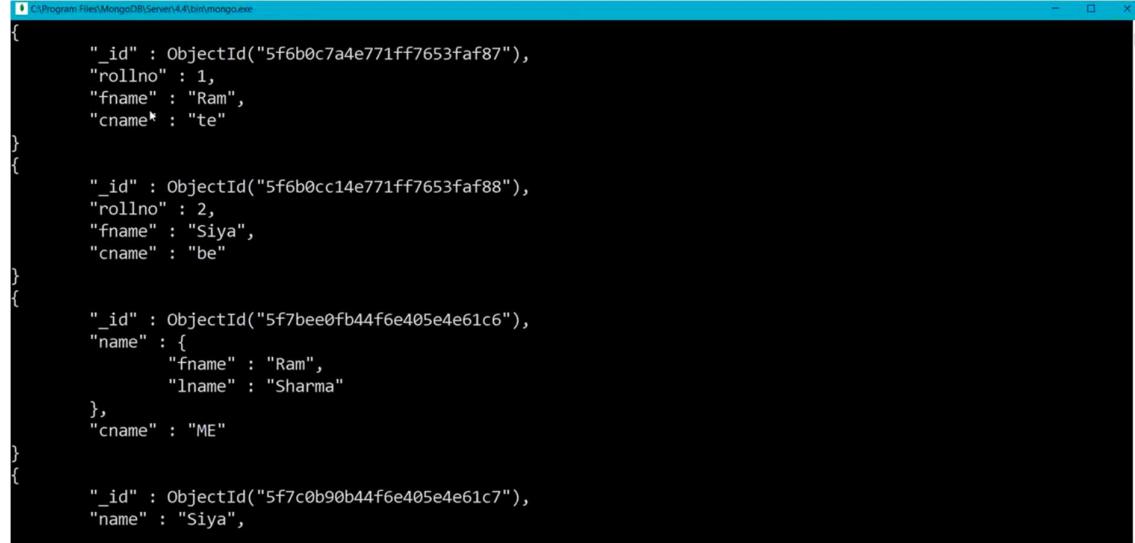
Demonstrate the indexing and ordering operations in MongoDB.

Solution :

Step 1 : Create Index for database.

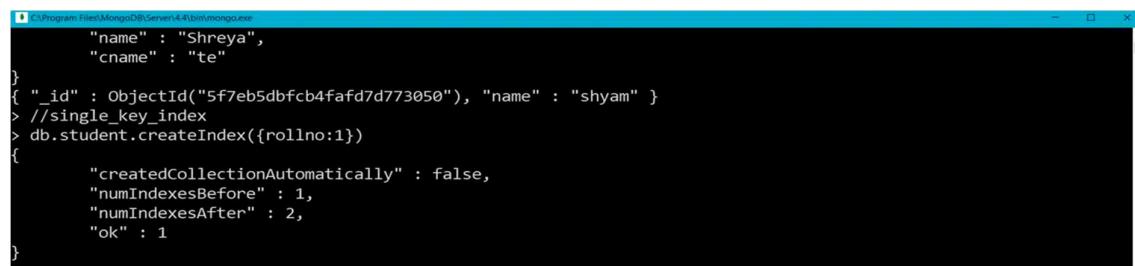


```
C:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
> //Indexing in Mongodb
> //Large collections,large documents
> db.student.find().pretty()
```

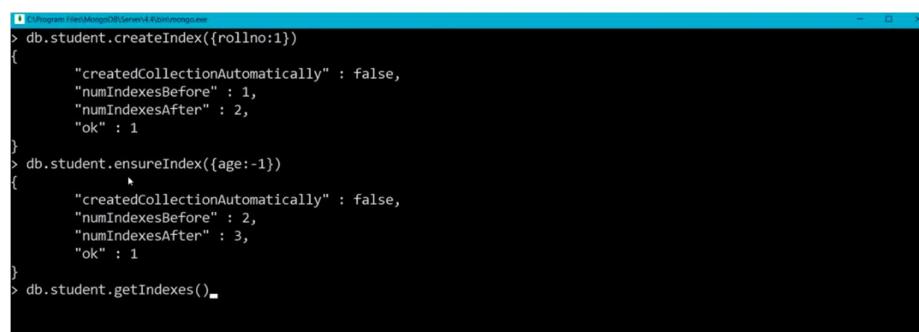
```
C:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
{
    "_id" : ObjectId("5f6b0c7a4e771ff7653faf87"),
    "rollno" : 1,
    "fname" : "Ram",
    "cname" : "te"
}
{
    "_id" : ObjectId("5f6b0cc14e771ff7653faf88"),
    "rollno" : 2,
    "fname" : "Siyा",
    "cname" : "be"
}
{
    "_id" : ObjectId("5f7bee0fb44f6e405e4e61c6"),
    "name" : {
        "fname" : "Ram",
        "lname" : "Sharma"
    },
    "cname" : "ME"
}
{
    "_id" : ObjectId("5f7c0b90b44f6e405e4e61c7"),
    "name" : "Siyा",
}
```

Step 2: For Single key index

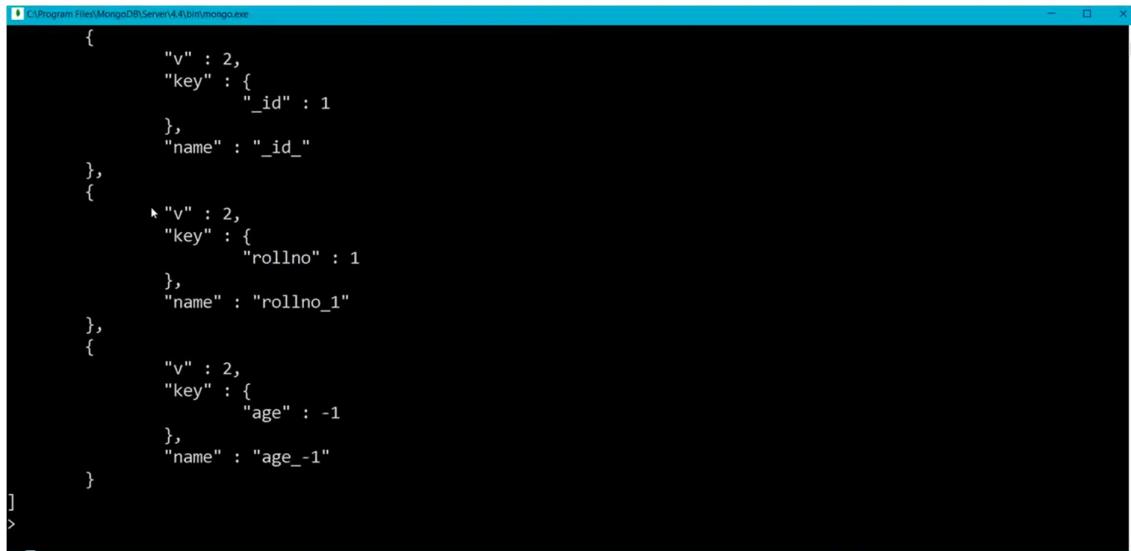


```
C:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
{
    "name" : "Shreya",
    "cname" : "te"
}
{
    "_id" : ObjectId("5f7eb5dbfc4fafd7d773050"),
    "name" : "shyam"
}
> //single_key_index
> db.student.createIndex({rollno:1})
{
    "createdCollectionAutomatically" : false,
    "numIndexesBefore" : 1,
    "numIndexesAfter" : 2,
    "ok" : 1
}
```

Another method to create index is Ensure index.



```
C:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
> db.student.createIndex({rollno:1})
{
    "createdCollectionAutomatically" : false,
    "numIndexesBefore" : 1,
    "numIndexesAfter" : 2,
    "ok" : 1
}
> db.student.ensureIndex({age:-1})
{
    "createdCollectionAutomatically" : false,
    "numIndexesBefore" : 2,
    "numIndexesAfter" : 3,
    "ok" : 1
}
> db.student.getIndexes()
```

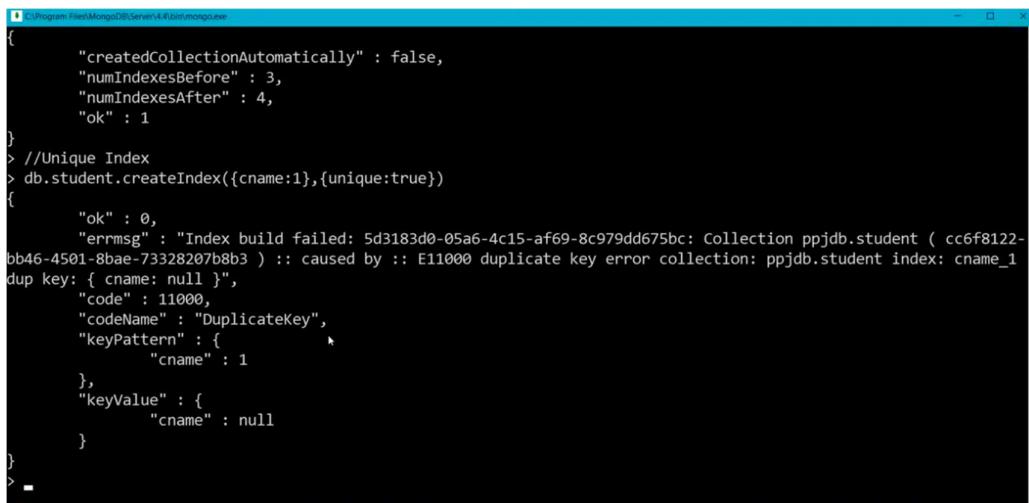


```
C:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
{
    "v" : 2,
    "key" : {
        "_id" : 1
    },
    "name" : "_id_"
},
{
    "v" : 2,
    "key" : {
        "rollno" : 1
    },
    "name" : "rollno_1"
},
{
    "v" : 2,
    "key" : {
        "age" : -1
    },
    "name" : "age_-1"
}
]
>
```

Step 3 : For compound key index.

```
] //compound_key_index
> db.student.createIndex({fname:1,lname:1})
{
    "createdCollectionAutomatically" : false,
    "numIndexesBefore" : 3,
    "numIndexesAfter" : 4,
    "ok" : 1
}
> -
```

Step 4 : For Unique key index.



```
{ "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 3,
  "numIndexesAfter" : 4,
  "ok" : 1
}
> //Unique Index
> db.student.createIndex({cname:1},{unique:true})
{
    "ok" : 0,
    "errmsg" : "Index build failed: 5d3183d0-05a6-4c15-af69-8c979dd675bc: Collection ppjdb.student ( cc6f8122-bb46-4501-8bae-73328207b8b3 ) :: caused by :: E11000 duplicate key error collection: ppjdb.student index: cname_1
dup key: { cname: null }",
    "code" : 11000,
    "codeName" : "DuplicateKey",
    "keyPattern" : {
        "cname" : 1
    },
    "keyValue" : {
        "cname" : null
    }
}
> -
```

```
12:30:00 AM Mon Dec 14 2015 [MongoDB:studentnew]
{
    "code" : 11000,
    "codeName" : "DuplicateKey",
    "keyPattern" : {
        "fname" : 1
    },
    "keyValue" : {
        "fname" : null
    }
}
> db.studentnew.find()
{ "_id" : ObjectId("5f9605fb8195e5f0e2d97816"), "name" : "Ram" }
{ "_id" : ObjectId("5f9606078195e5f0e2d97817"), "name" : "Siya" }
> db.studentnew.createIndex({name:1},{unique:true})
{
    "createdCollectionAutomatically" : false,
    "numIndexesBefore" : 1,
    "numIndexesAfter" : 2,
    "ok" : 1
}
> -
```

Practical No 7

Demonstrating Map reduce in MongoDB.

Solution:

Step 1 : Create Data collection

```
C:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
> //MapReduce in MongoDB
> //Map
> //Reduce
> db.studentdata.find()
{ "_id" : ObjectId("5f94ae8e78b6fd018d4b9448"), "name" : "Ram", "marks" : 40, "age" : 20 }
{ "_id" : ObjectId("5f94ae8778b6fd018d4b9449"), "name" : "Sya", "marks" : 50, "age" : 20 }
{ "_id" : ObjectId("5f94aec78b6fd018d4b944a"), "name" : "Swati", "marks" : 30, "age" : 18 }
{ "_id" : ObjectId("5f94aee178b6fd018d4b944b"), "name" : "Rahul", "marks" : 50, "age" : 21 }
{ "_id" : ObjectId("5f94aef678b6fd018d4b944c"), "name" : "Riya", "marks" : 60, "age" : 21 }
> //calculate the sum of marks of all students
>
```

Step 2 : To perform Map reduce function.

```
C:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
> //MapReduce in MongoDB
> //Map
> //Reduce
> db.studentdata.find()
{ "_id" : ObjectId("5f94ae8e78b6fd018d4b9448"), "name" : "Ram", "marks" : 40, "age" : 20 }
{ "_id" : ObjectId("5f94ae8778b6fd018d4b9449"), "name" : "Sya", "marks" : 50, "age" : 20 }
{ "_id" : ObjectId("5f94aec78b6fd018d4b944a"), "name" : "Swati", "marks" : 30, "age" : 18 }
{ "_id" : ObjectId("5f94aee178b6fd018d4b944b"), "name" : "Rahul", "marks" : 50, "age" : 21 }
{ "_id" : ObjectId("5f94aef678b6fd018d4b944c"), "name" : "Riya", "marks" : 60, "age" : 21 }
> //calculate the sum of marks of all students
> var mapfunction=function(){emit(this.age,this.marks)}
> var reducefunction=function(key,values){return Array.sum(values)}
> db.studentdata.mapReduce(mapfunction,reducefunction,['out':'Result1_mapreduce'])
{ "result" : "Result1_mapreduce", "ok" : 1 }
> db.Result1_mapreduce.find()
{ "_id" : 18, "value" : 30 }
{ "_id" : 21, "value" : 110 }
{ "_id" : 20, "value" : 90 }
>
```

```
C:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
> //MapReduce in MongoDB
> //Map
> //Reduce
> db.studentdata.find()
{ "_id" : ObjectId("5f94ae8e78b6fd018d4b9448"), "name" : "Ram", "marks" : 40, "age" : 20 }
{ "_id" : ObjectId("5f94ae8778b6fd018d4b9449"), "name" : "Sya", "marks" : 50, "age" : 20 }
{ "_id" : ObjectId("5f94aec78b6fd018d4b944a"), "name" : "Swati", "marks" : 30, "age" : 18 }
{ "_id" : ObjectId("5f94aee178b6fd018d4b944b"), "name" : "Rahul", "marks" : 50, "age" : 21 }
{ "_id" : ObjectId("5f94aef678b6fd018d4b944c"), "name" : "Riya", "marks" : 60, "age" : 21 }
> //calculate the sum of marks of all students
> var mapfunction=function(){emit(this.age,this.marks)}
> var reducefunction=function(key,values){return Array.sum(values)}
> db.studentdata.mapReduce(mapfunction,reducefunction,['out':'Result1_mapreduce'])
{ "result" : "Result1_mapreduce", "ok" : 1 }
> db.Result1_mapreduce.find()
{ "_id" : 18, "value" : 30 }
{ "_id" : 21, "value" : 110 }
{ "_id" : 20, "value" : 90 }
> db.studentdata.mapReduce(function(){emit(this.age,this.marks)},function(key,values){
... return Array.avg(values)},{'query:{age:{>:18}},out:'Result2_mapreduce'})
{ "result" : "Result2_mapreduce", "ok" : 1 }
> db.Result2_mapreduce.find()
```

Practical No 8

Demonstrate the Accessing and Storing and performing CRUD operations in Apache Cassandra.

Solution :

Step 1 : Create Record

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure under the "Cassandra" folder, including JRE System Library [JavaSE-1.8] and various dependency jars like cassandra-driver-core, cassandra-driver-dse, cassandra-driver-mapping, etc.
- Code Editor:** Displays the `GettingStarted.java` file with Java code for connecting to a Cassandra cluster and inserting a record into the "users" table.
- Console:** Shows the output of the Java application, indicating successful insertion of a user named "test" with age 15.
- Task List:** Shows a task related to "GettingStarted" with the message "Update the same user with a new age".
- Outline:** Shows the class structure with the `main(String[] args)` method.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure under the "Cassandra" folder, including JRE System Library [JavaSE-1.8] and various dependency jars.
- Code Editor:** Displays the `GettingStarted.java` file with Java code for connecting to a Cassandra cluster and inserting a record into the "users" table.
- Console:** Shows the output of the Java application, indicating successful insertion of a user named "test" with age 15.
- Task List:** Shows a task related to "GettingStarted" with the message "Update the same user with a new age".
- Outline:** Shows the class structure with the `main(String[] args)` method.
- CQL Shell:** A separate window shows the results of a CQL query: `cqlsh:demo> select * from users;`. The output shows a single row: `lastname | age | city | email | firstname` with values `test | 15 | tampa | test@123.com | test1`.

Step 2 : To insert record.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with a package named `cassandra_tester.java`.
- Code Editor:** Displays Java code for `GettingStarted.java`. The code includes imports for `com.datastax.driver.core.Cluster`, `com.datastax.driver.core.ResultSet`, `com.datastax.driver.core.Row`, and `com.datastax.driver.core.Session`. It defines a class `GettingStarted` with a static main method that creates a cluster, connects to a "demo" keyspace, and inserts a record into the "users" table. The record has "lastname" as "chumma", "age" as 35, "city" as "tampa", "email" as "chumma@example.com", and "firstname" as "test".

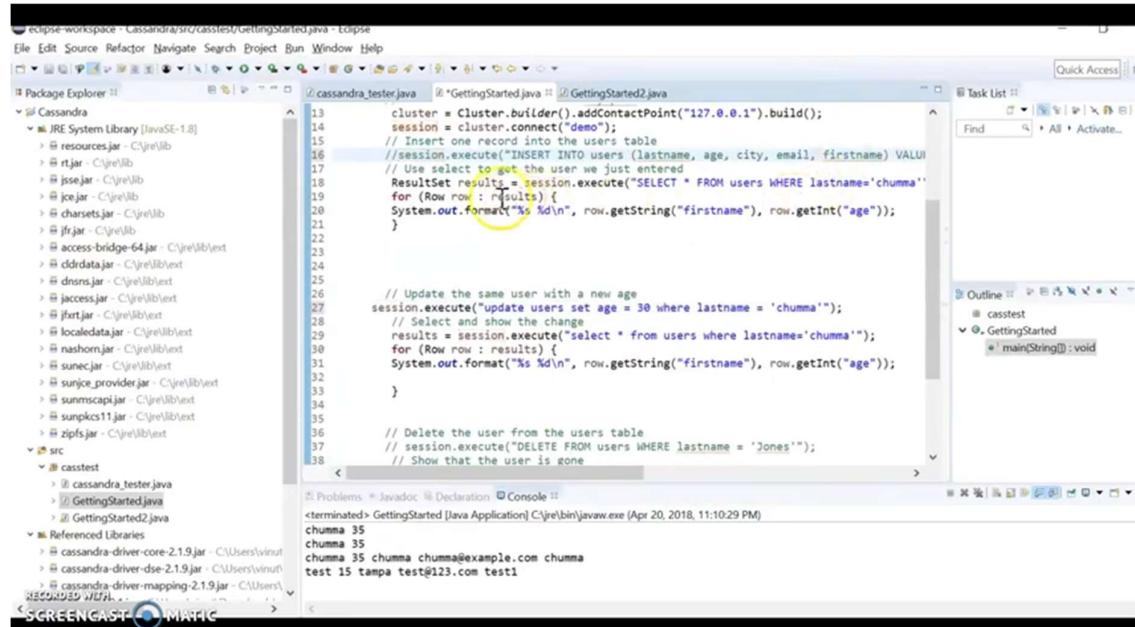
```
1 package casstest;
2
3 import com.datastax.driver.core.Cluster;
4 import com.datastax.driver.core.ResultSet;
5 import com.datastax.driver.core.Row;
6 import com.datastax.driver.core.Session;
7
8 public class GettingStarted {
9     public static void main(String[] args){
10         Cluster cluster;
11         Session session;
12         // Connect to the cluster and keyspace "demo"
13         cluster = Cluster.builder().addContactPoint("127.0.0.1").build();
14         session = cluster.connect("demo");
15         // Insert one record into the users table
16         session.execute("INSERT INTO users (lastname, age, city, email, firstname) VALUES
17             // Use select to get the user we just entered
18             ResultsSet results = session.execute("SELECT * FROM users WHERE lastname='chumma'");
19             for (Row row : results) {
20                 System.out.format("%s %d\n", row.getString("firstname"), row.getInt("age"));
21             }
22
23
24
25         // Update the same user with a new age
26     }
27 }
```
- Task List:** Shows a single task named `GettingStarted`.
- Outline:** Shows the class `GettingStarted` and its `main` method.
- Console:** Displays the output of the Java application, showing the inserted record and the updated record.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with a package named `cassandra_tester.java`.
- Cassandra CQL Shell:** A separate window showing the results of a CQL query. The query `select * from users;` was run twice, once after the insertion and once after the update. The results show two rows: the first row has "lastname" as "chumma", "age" as 35, "city" as "tampa", "email" as "chumma@example.com", and "firstname" as "test1". The second row has "lastname" as "chumma", "age" as 30, "city" as "chumma", "email" as "chumma@example.com", and "firstname" as "chumma".

lastname	age	city	email	firstname
chumma	35	tampa	chumma@example.com	test1
chumma	30	chumma	chumma@example.com	chumma
- Task List:** Shows a single task named `GettingStarted`.
- Outline:** Shows the class `GettingStarted` and its `main` method.

Step 3 : To update Record.

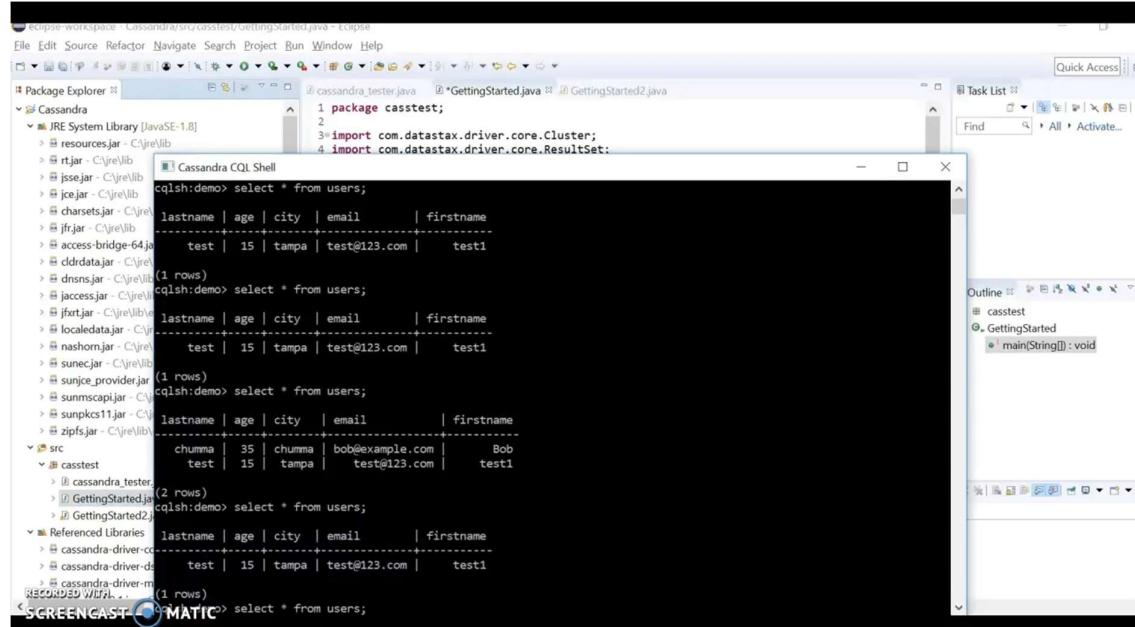


The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with a **Cassandra** folder containing JRE System Library [JavaSE-1.8] and Reference Libraries.
- Editor:** Displays the `cassandra_tester.java` file containing Java code for interacting with a Cassandra database. A yellow circle highlights the line `System.out.format("%s %d\n", row.getString("firstname"), row.getInt("age"));`.


```

13     cluster = Cluster.builder().addContactPoint("127.0.0.1").build();
14     session = cluster.connect("demo");
15     // Insert one record into the users table
16     //session.execute("INSERT INTO users (lastname, age, city, email, firstname) VALUES
17     // Use select to get the user we just entered
18     ResultSet results = session.execute("SELECT * FROM users WHERE lastname='chumma'");
19     for (Row row : results) {
20         System.out.format("%s %d\n", row.getString("firstname"), row.getInt("age"));
21     }
22
23
24
25
26     // Update the same user with a new age
27     session.execute("update users set age = 30 where lastname = 'chumma'");
28     // Select and show the change
29     results = session.execute("select * from users where lastname='chumma'");
30     for (Row row : results) {
31         System.out.format("%s %d\n", row.getString("firstname"), row.getInt("age"));
32     }
33
34
35     // Delete the user from the users table
36     // session.execute("DELETE FROM users WHERE lastname = 'Jones'");
37     // Show that the user is gone
      
```
- Console:** Shows the output of the Java application running in the terminal, confirming the insertion, update, and deletion of a user record.

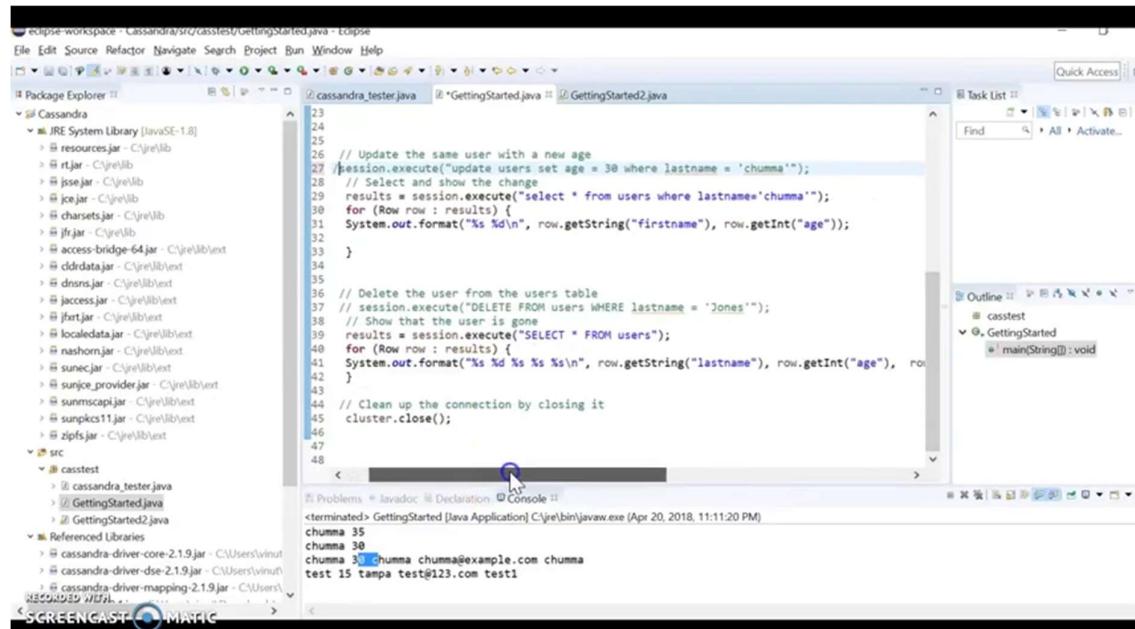


The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with a **Cassandra** folder containing JRE System Library [JavaSE-1.8] and Reference Libraries.
- Editor:** Displays the `cassandra_tester.java` file containing Java code for interacting with a Cassandra database.
- CQL Shell:** A separate window titled "Cassandra CQL Shell" is open, showing the results of CQL queries. It displays two rows of data from the "users" table:

lastname	age	city	email	firstname
test	15	tampa	test@123.com	test1
chumma	35	chumma	bob@example.com	Bob
- Console:** Shows the output of the Java application running in the terminal, confirming the insertion, update, and deletion of a user record.

Step 4 : To delete record.



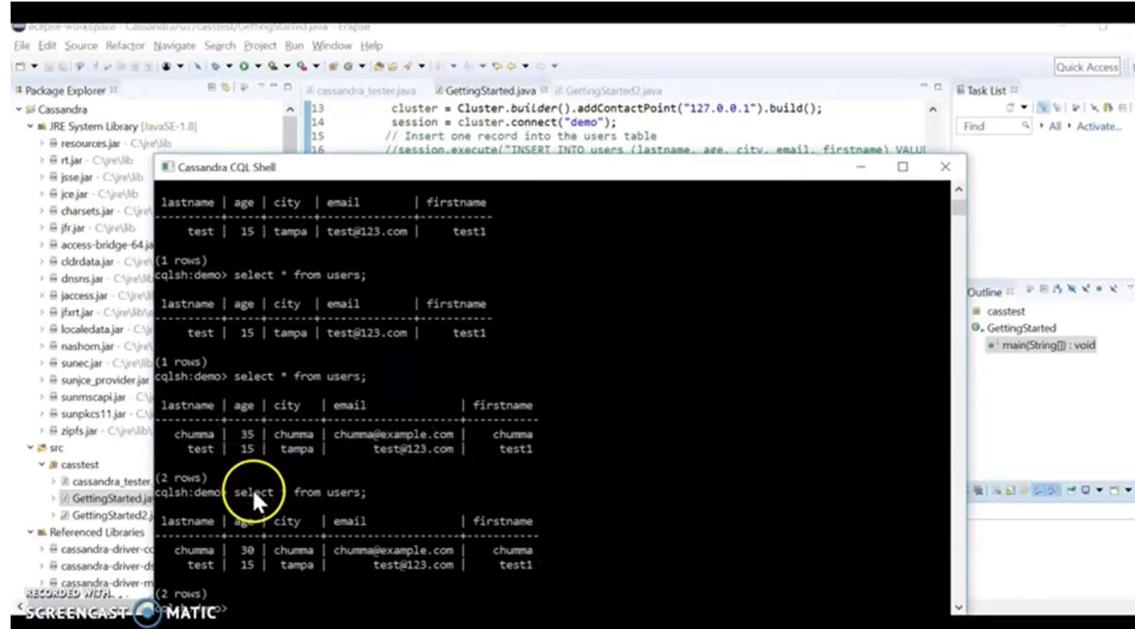
The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with a **Cassandra** folder containing JRE System Library [JavaSE-1.8] and Reference Libraries.
- Code Editor:** Displays `cassandra_tester.java` with Java code for updating and deleting a user record. The code uses the `cassandra-driver-core-2.1.9.jar` library.
- Console:** Shows the output of the Java application execution, displaying the updated user record and the deletion process.
- Outline:** Shows the class structure with a `main(String[])` method.

```

eclipse-workspace - Cassandra/src/cassetest/GettingStarted.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer GettingStarted.java GettingStarted2.java
cassandra_tester.java
23
24
25
26 // Update the same user with a new age
27 session.execute("update users set age = 30 where lastname = 'chumma'");
28 // Select and show the change
29 results = session.execute("select * from users where lastname='chumma'");
30 for (Row row : results) {
31     System.out.format("%s %d\n", row.getString("firstname"), row.getInt("age"));
32 }
33
34
35
36 // Delete the user from the users table
37 session.execute("DELETE FROM users WHERE lastname = 'Jones'");
38 // Show that the user is gone
39 results = session.execute("SELECT * FROM users");
40 for (Row row : results) {
41     System.out.format("%s %d %s %s\n", row.getString("lastname"), row.getInt("age"), row.getString("city"), row.getString("email"));
42 }
43
44 // Clean up the connection by closing it
45 cluster.close();
46
47
48

```



The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with a **Cassandra** folder containing JRE System Library [JavaSE-1.8] and Reference Libraries.
- Cassandra CQL Shell:** A separate window showing the CQL shell interface. It displays the insertion of a new user record and subsequent SELECT statements showing the data.
- Outline:** Shows the class structure with a `main(String[])` method.

```

lastname | age | city | email | firstname
test | 15 | tampa | test@123.com | test1
(1 rows)
cqsh:demo> select * from users;
lastname | age | city | email | firstname
test | 15 | tampa | test@123.com | test1
(1 rows)
cqsh:demo> select * from users;
lastname | age | city | email | firstname
chumma | 35 | chumma | chumma@example.com | chumma
test | 15 | tampa | test@123.com | test1
(2 rows)
cqsh:demo> select * from users;
lastname | age | city | email | firstname
chumma | 30 | chumma | chumma@example.com | chumma
test | 15 | tampa | test@123.com | test1
(2 rows)

```