**Uploading the data:**

**package** com.samp.proj;

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.util.Arrays;

**import** com.google.api.client.googleapis.auth.oauth2.GoogleAuthorizationCodeFlow;

**import** com.google.api.client.googleapis.auth.oauth2.GoogleCredential;

**import** com.google.api.client.googleapis.auth.oauth2.GoogleTokenResponse;

**import** com.google.api.client.http.FileContent;

**import** com.google.api.client.http.HttpTransport;

**import** com.google.api.client.http.javanet.NetHttpTransport;

**import** com.google.api.client.json.JsonFactory;

**import** com.google.api.client.json.jackson2.JacksonFactory;

**import** com.google.api.services.drive.Drive;

**import** com.google.api.services.drive.DriveScopes;

**import** com.google.api.services.drive.model.File;

**public** **class** sampproj {

//Cloud client ID and secret

**private** **static** String *CLIENT\_ID* = "80356728864-h250fm8jo6chc22k7p7l9kklj3v3gond.apps.googleusercontent.com";

**private** **static** String *CLIENT\_SECRET* = "MW5sHUm\_Qeuv4V5lEWflZI69";

//Redirect URI for the auth codce

**private** **static** String *REDIRECT\_URI* = "urn:ietf:wg:oauth:2.0:oob";

**public** **static** **void** main(String[] args) **throws** IOException {

HttpTransport httpTransport = **new** NetHttpTransport();

JsonFactory jsonFactory = **new** JacksonFactory();

//Authentication Flow

GoogleAuthorizationCodeFlow flow = **new** GoogleAuthorizationCodeFlow.Builder(

httpTransport, jsonFactory, *CLIENT\_ID*, *CLIENT\_SECRET*, Arrays.*asList*(DriveScopes.*DRIVE*))

.setAccessType("online")

.setApprovalPrompt("auto").build();

String url = flow.newAuthorizationUrl().setRedirectUri(*REDIRECT\_URI*).build();

System.*out*.println("Please open the following URL in your browser then type the authorization code:");

System.*out*.println(" " + url);

BufferedReader br = **new** BufferedReader(**new** InputStreamReader(System.*in*));

String code = br.readLine();

GoogleTokenResponse response = flow.newTokenRequest(code).setRedirectUri(*REDIRECT\_URI*).execute();

GoogleCredential credential = **new** GoogleCredential().setFromTokenResponse(response);

//Create a new authorized API client

Drive service = **new** Drive.Builder(httpTransport, jsonFactory, credential).build();

//-----------------------------------

//Insert 1st file -- size 20kb

//Calculation Upload time

**long** time1=System.*currentTimeMillis*();

**long** sec1=time1/1000;

File body = **new** File();

body.setTitle("Cloudproj4 doc");

body.setDescription("A test document");

body.setMimeType("text/plain");

java.io.File fileContent = **new** java.io.File("cloudproj4.txt");

FileContent mediaContent = **new** FileContent("text/plain", fileContent);

File file = service.files().insert(body, mediaContent).execute();

System.*out*.println("File ID: " + file.getId());

System.*out*.println("File ID: " + file.getFileSize());

System.*out*.println("File ID: " + file.getDownloadUrl());

**long** time2=System.*currentTimeMillis*();

**long** sec2=time2/1000;

**long** timeTaken=sec2-sec1;

System.*out*.println("Total time taken for two files "+ timeTaken +" seconds");

//-----------------------

//Insert 2nd file -- size 60kb

**long** time3=System.*currentTimeMillis*();

**long** sec3=time3/1000;

File body1 = **new** File();

body1.setTitle("Cldinp doc");

body1.setDescription("A test document");

body1.setMimeType("text/plain");

java.io.File fileContent1 = **new** java.io.File("cldinp.csv");

FileContent mediaContent1 = **new** FileContent("text/plain", fileContent1);

File file1 = service.files().insert(body1, mediaContent1).execute();

System.*out*.println("File ID: " + file1.getId());

System.*out*.println("File ID: " + file1.getFileSize());

System.*out*.println("File ID: " + file1.getDownloadUrl());

**long** time4=System.*currentTimeMillis*();

**long** sec4=time4/1000;

**long** timeTaken1=sec4-sec3;

System.*out*.println("Total time taken for two files "+ timeTaken1 +" seconds");

//---------------------------------------

//---------------------------------------

//Insert 3rd file -- size 100kb

**long** time5=System.*currentTimeMillis*();

**long** sec5=time5/1000;

File body2 = **new** File();

body2.setTitle("6331 doc");

body2.setDescription("A test document");

body2.setMimeType("text/plain");

java.io.File fileContent2 = **new** java.io.File("6331.csv");

FileContent mediaContent2 = **new** FileContent("text/plain", fileContent2);

File file2 = service.files().insert(body2, mediaContent2).execute();

System.*out*.println("File ID: " + file2.getId());

System.*out*.println("File ID: " + file2.getFileSize());

System.*out*.println("File ID: " + file2.getDownloadUrl());

**long** time6=System.*currentTimeMillis*();

**long** sec6=time6/1000;

**long** timeTaken2=sec6-sec5;

System.*out*.println("Total time taken for two files "+ timeTaken2 +" seconds");

//---------------------------------------

}

}

**Downloading the data:**

**package** com.samp.proj;

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.io.InputStreamReader;

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.List;

**import** com.google.api.client.googleapis.auth.oauth2.GoogleAuthorizationCodeFlow;

**import** com.google.api.client.googleapis.auth.oauth2.GoogleCredential;

**import** com.google.api.client.googleapis.auth.oauth2.GoogleTokenResponse;

**import** com.google.api.client.http.GenericUrl;

**import** com.google.api.client.http.HttpResponse;

**import** com.google.api.client.http.HttpTransport;

**import** com.google.api.client.http.javanet.NetHttpTransport;

**import** com.google.api.client.json.JsonFactory;

**import** com.google.api.client.json.jackson2.JacksonFactory;

**import** com.google.api.services.drive.Drive;

**import** com.google.api.services.drive.Drive.Files;

**import** com.google.api.services.drive.DriveScopes;

**import** com.google.api.services.drive.model.File;

**import** com.google.api.services.drive.model.FileList;

// ...

**public** **class** storeproj {

// ...

/\*\*

\* Download a file's content.

\*

\* **@param** service Drive API service instance.

\* **@param** file Drive File instance.

\* **@return** InputStream containing the file's content if successful,

\* {@code null} otherwise.

\*/

**private** **static** String *CLIENT\_ID* = "80356728864-h250fm8jo6chc22k7p7l9kklj3v3gond.apps.googleusercontent.com";

**private** **static** String *CLIENT\_SECRET* = "MW5sHUm\_Qeuv4V5lEWflZI69";

//Redirect URI for the auth codce

**private** **static** String *REDIRECT\_URI* = "urn:ietf:wg:oauth:2.0:oob";

**private** **static** InputStream downloadFile(Drive service, File file) {

**if** (file.getDownloadUrl() != **null** && file.getDownloadUrl().length() > 0) {

**try** {

HttpResponse resp =

service.getRequestFactory().buildGetRequest(**new** GenericUrl(file.getDownloadUrl()))

.execute();

**return** resp.getContent();

} **catch** (IOException e) {

// An error occurred.

e.printStackTrace();

**return** **null**;

}

} **else** {

// The file doesn't have any content stored on Drive.

**return** **null**;

}

}

**private** **static** List<File> retrieveAllFiles(Drive service) **throws** IOException {

List<File> result = **new** ArrayList<File>();

Files.List request = service.files().list();

**do** {

**try** {

FileList files = request.execute();

result.addAll(files.getItems());

request.setPageToken(files.getNextPageToken());

} **catch** (IOException e) {

System.*out*.println("An error occurred: " + e);

request.setPageToken(**null**);

}

} **while** (request.getPageToken() != **null** &&

request.getPageToken().length() > 0);

**return** result;

}

// ...

**public** **static** **void** main(String[] args) **throws** IOException {

HttpTransport httpTransport = **new** NetHttpTransport();

JsonFactory jsonFactory = **new** JacksonFactory();

//Authentication Flow

GoogleAuthorizationCodeFlow flow = **new** GoogleAuthorizationCodeFlow.Builder(

httpTransport, jsonFactory, *CLIENT\_ID*, *CLIENT\_SECRET*, Arrays.*asList*(DriveScopes.*DRIVE*))

.setAccessType("online")

.setApprovalPrompt("auto").build();

String url = flow.newAuthorizationUrl().setRedirectUri(*REDIRECT\_URI*).build();

System.*out*.println("Please open the following URL in your browser then type the authorization code:");

System.*out*.println(" " + url);

BufferedReader br = **new** BufferedReader(**new** InputStreamReader(System.*in*));

String code = br.readLine();

GoogleTokenResponse response = flow.newTokenRequest(code).setRedirectUri(*REDIRECT\_URI*).execute();

GoogleCredential credential = **new** GoogleCredential().setFromTokenResponse(response);

//Create a new authorized API client

Drive service = **new** Drive.Builder(httpTransport, jsonFactory, credential).build();

List<File> file = *retrieveAllFiles*(service);

**long** time1=System.*currentTimeMillis*();

**long** sec1=time1/1000;

downloadFile(service, file);

**long** time2=System.*currentTimeMillis*();

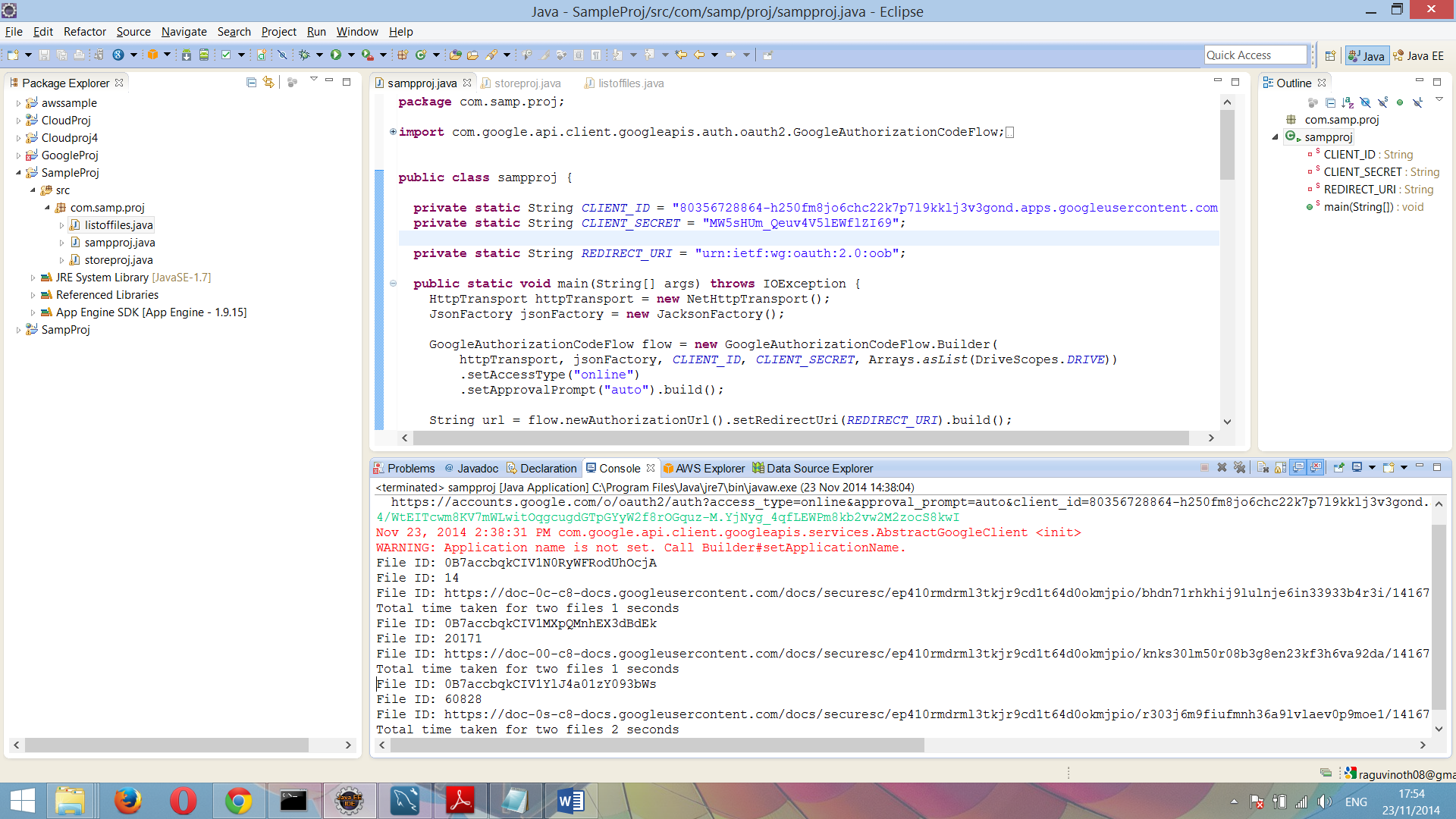
**long** sec2=time2/1000;

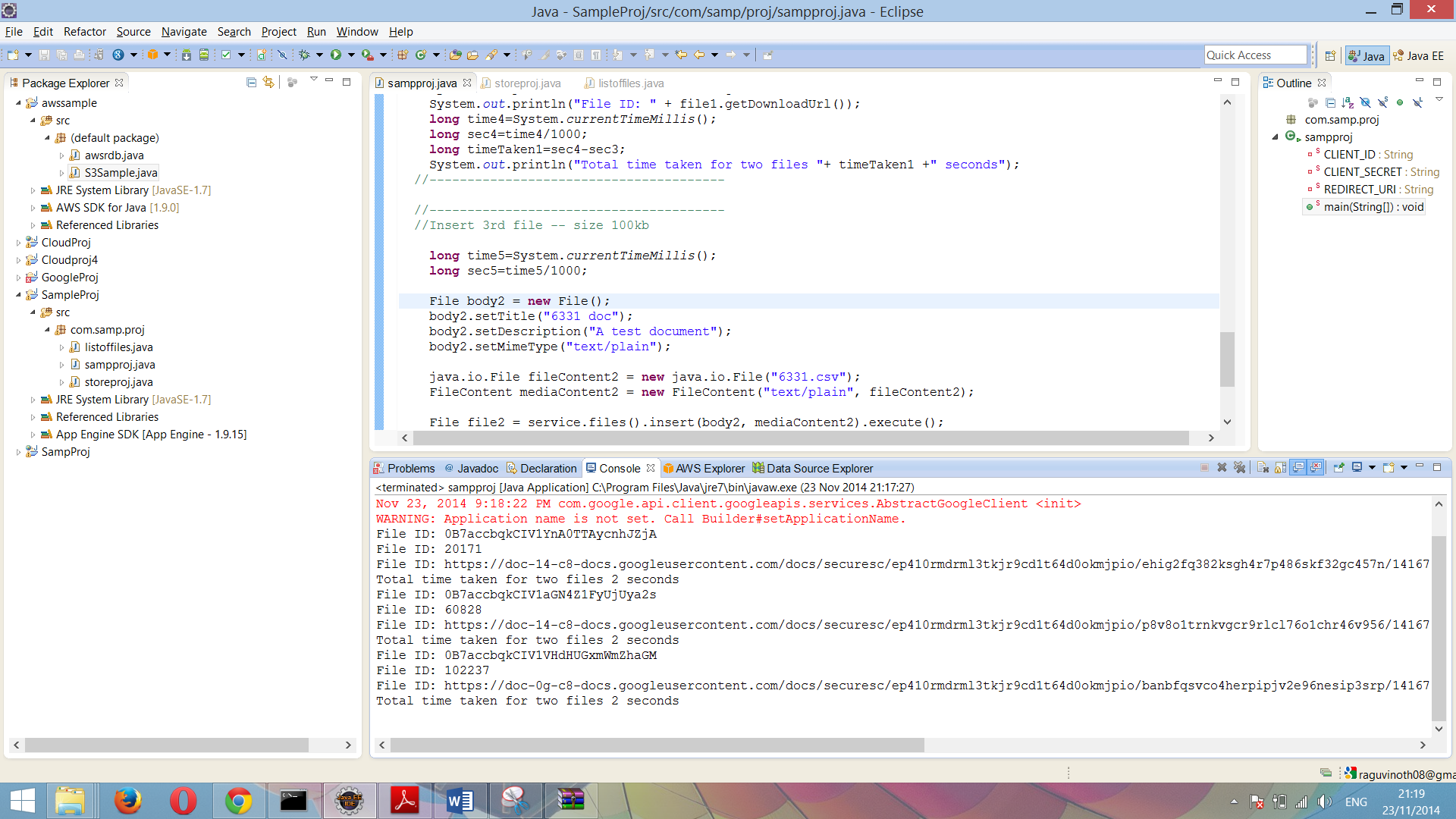
**long** timeTaken=sec2-sec1;

System.*out*.println("Total time taken for two files "+ timeTaken +" seconds");

}

}





Cloud computing performance can be measured in several ways. Common metrics include:

* **Uptime and Availability**—In terms of business and operation continuity, consider factors that influence your cloud’s uptime, such as scalability, failover and backup systems.
* **Speed** -- How fast will applications launch? What input/output operations per second (IOPS) does your business operate with currently, and what does it need to perform better? Ask about promised computing speeds, and the technologies behind them.
* **The bandwidth of the network**

In the first screenshot, I have uploaded three files of size 10kb, 20kb and 60kb of data which takes about 1 second, 1 second and 2 seconds respectively.

And in the second screenshot, I have uploaded three files of size 20kb, 60kb and 100kb of the data which takes about 2 seconds each respectively.

**Google SQL – MySQL Workbench:**

create database cloudsql;

use cloudsql;

create table sqlcloud

(

AGE varchar(10),

SEX varchar(10),

RACE varchar(10),

DAY\_OF\_ADMISSION varchar(10),

DISCHARGE\_STATUS varchar(10),

STAY\_INDICATOR varchar(10),

DRG\_CODE varchar(10),

LENGTH\_OF\_STAY varchar(10),

DRG\_PRICE varchar(10),

TOTAL\_CHARGES varchar(10),

COVERED\_CHARGES varchar(10),

POA\_DIAGNOSIS\_INDICATOR\_1 varchar(10),

POA\_DIAGNOSIS\_INDICATOR\_2 varchar(10),

DIAGNOSIS\_CODE\_1 varchar(10),

DIAGNOSIS\_CODE\_2 varchar(10),

PROCEDURE\_CODE\_1 varchar(10),

PROCEDURE\_CODE\_2 varchar(10),

DISCHARGE\_DESTINATION varchar(10),

SOURCE\_OF\_ADMISSION varchar(10),

TYPE\_OF\_ADMISSION varchar(10),

ADMITTING\_DIAGNOSIS\_CODE varchar(10)

);

select \* from sqlcloud;

LOAD DATA LOCAL INFILE 'C:/Users/Raguvinoth/Desktop/dbproj/6331.csv'

INTO TABLE sqlcloud

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

SELECT AGE AS AGE\_GROUP, (SUM(LENGTH\_OF\_STAY)/count(\*)) AS MEAN\_LOS

FROM sqlcloud

GROUP BY AGE;

