Informatics Institute of technology

In collaboration with

University of Westminster, UK

# Information Systems and Business management

Business Intelligence

6BUIS001 W

**Coursework 1**

D.M.V. Oshadhi……...….: 2015078

# Table of Contents

# 1. DESIGNING A SINGLE FACT SCHEMA FOR A HOTEL CHAIN

## a) BUSINESS PROCESSORS/ LOGICAL SCHEMA

BOOKINGS (hotel_id*, time_id*, location_id*, room_no, status, expected_income)

CHECKOUT (hotel_id*, time_id*, location_id*, feature_id*, room_no, income)

TABLE VIEW OF LOGICAL SCHEMA

| Bookings |
| --- |
| +hotel_id {PK}{FK} |
| +time_id {PK}{FK} |
| +location_id {PK}{FK} |
| +room_no {PK} |
| + status |
| + expected_income |

| Checkout |
| --- |
| +hotel_id {PK}{FK} |
| +time_id {PK}{FK} |
| +location_id {PK}{FK} |
| +feature_id {PK}{FK} |
| +room_no {PK} |
| + income |

JUSTIFICATION

Fact table is a table which is referred to as a table which joins dimension tables with measures in data warehousing theory. One of the key features is these particular attributes, named as Measures containing numerical data could be manipulated to derive a meaningful result and will assist higher managers to make informed business decisions. In the given case scenario of hotel chain, the main business processors were identified as Bookings and Checkout, because they are the key main processors within a hotel which will assist to analyse the income earned or lost, how facilities provided are affecting the income earned, analyse potential hotels with higher revenues earned during past years to measure the success of objectives set by the hotel chain managers.

Thus, with the 2 identified business processors “Bookings” and “Checkout” as fact tables were identified as they are measurable according to this scenario. Moreover, Payments was not considered as a fact as customers spending habits will have a minimum impact to income earned in a hotel. Hence,

as Customer's Payments is immeasurable and income per room was already provided in source schema to analyse the income or the percentage of rooms it was not considered as a fact.

## b) DIMENSIONS AND THEIR LOGICAL SCHEMA

Hotel (hotel_id, hotel_name, category)

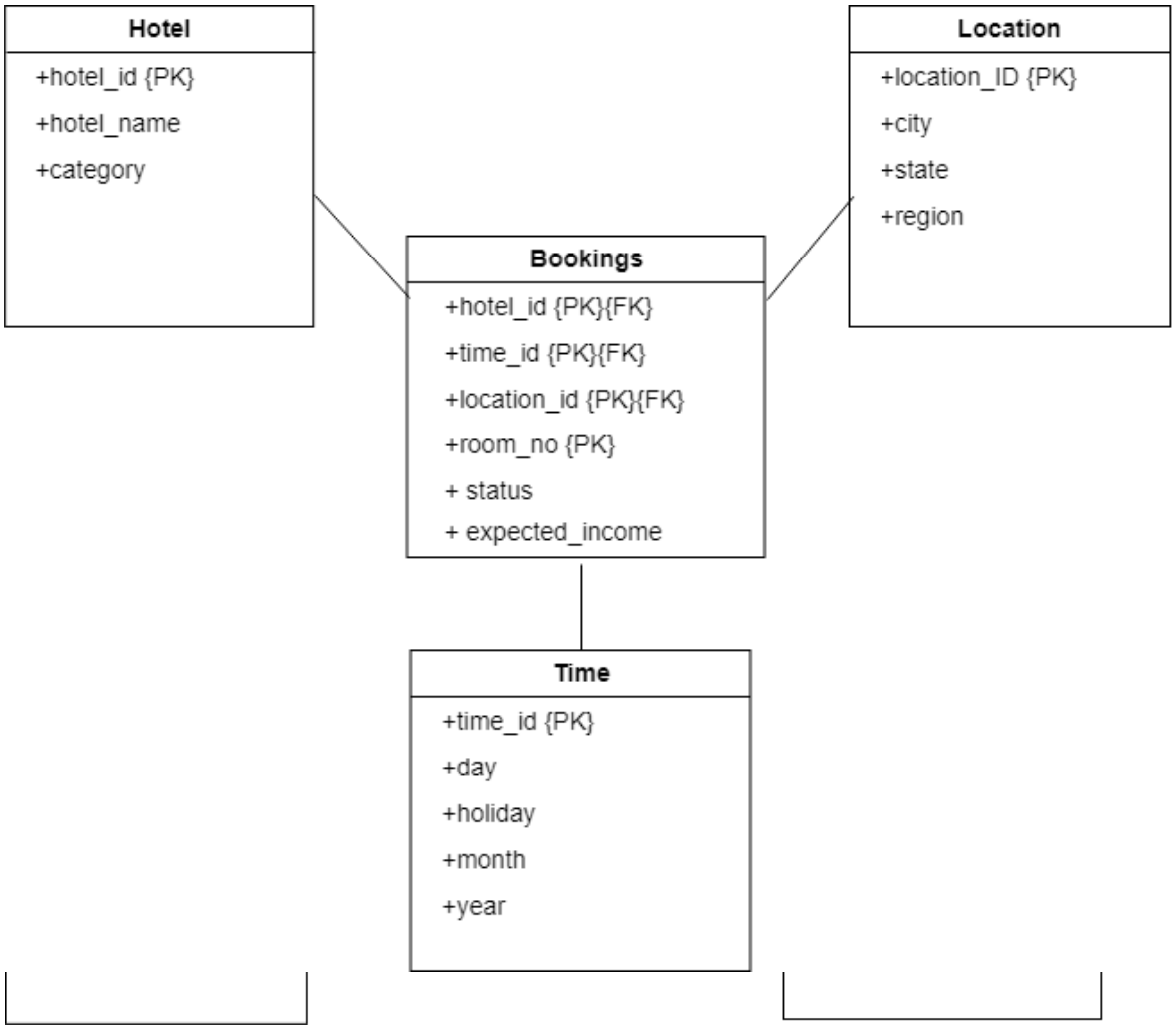
Location (location_id, city, state, region)

Time (time_id, day, holiday, month, year)]

Feature (feature_id, feature_description)

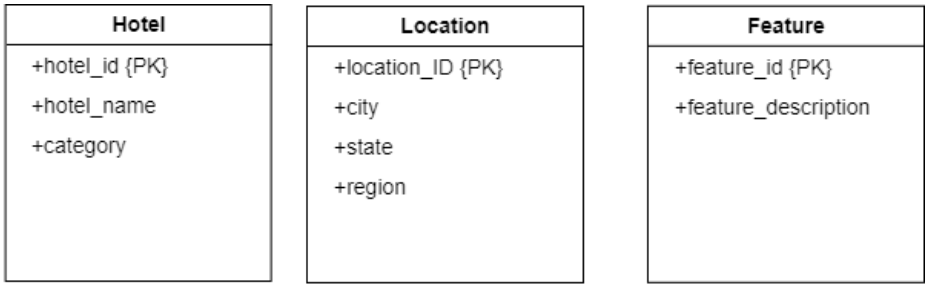
TABLE VIEW OF LOGICAL DATA

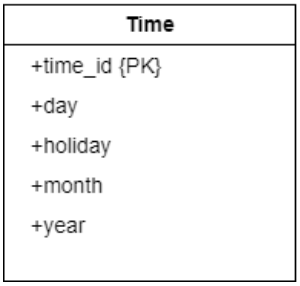
## RELATED DIMENSIONS FOR BOOKINGS (FACT) TABLE




## RELATED DIMENSIONS FOR CHECKOUT (FACT) TABLE

## SINGLE DIMENSION TABLES




JUSTIFICATION

Hotel, Time, Location, Feature are the identified dimensions. A dimension is a structure which categories facts and measures in order to enable business operations. In a data warehouse, dimensions provide structured labelling information to unordered numeric measures. The dimensions were identified by identifying through Who, What, Where and When concept. These dimensions are related to the fact tables and they are non-measurable.

## c) MEASURES FOR EACH BUSINESS PROCESSES

| Business Processors/ Facts | BOOKINGS FACT | CHECKOUT FACT |
|---|---|---|
| Measures | status<br>expected_income | income |
| Measures derived from | Status – Obtained from Bookings table<br>Expected Income – Obtained from Boookings table | Income – Obtained from Checkouts table as given in the source schema |

## JUSTIFICATION

Measures are data elements that can be mathematically manipulated and properties that could be used to make an informed business decision. Measures are numeric representations of a set of facts that have occurred. In this scenario following are the reasons for selecting the measures,

**Bookings** – The scenario explains that hotel chain managers would prefer to calculate the percentage of free rooms, reserved rooms, unavailable rooms. Hence, status of rooms was considered as a measure to calculate the individual percentages for three statuses of rooms which differ according to time of day, month or year and hotel location as there are no. of rooms within each hotel. Moreover, expected income of bookings was included as hotel chain managers would like to analyze the income of each hotel daily, monthly and yearly. By analyzing expected income they would be able to understand how with time bookings are affected(To plan marketing strategies if booking incomes are low), according to location how expected income varies due to reasons such as customer preferences.

Percentage was not identified as a measure as granularity improved when statuses of rooms were considered, and also percentage is the final result which needs to be calculated by manipulating the identified measures. Moreover, percentages of rooms could be calculated through bookings only as checkout will only compromise of records of rooms which the customer has checked in only. Thus, status and expected income will be a measure in Bookings facts table.

**Checkout** – Actual income earned is stored within checkouts as booking cancellations are omitted. Hence, managers would be able to analyze the income earned through daily, monthly or yearly check outs to analyze the income according to the geographical location, hotel category and room features. Thus, with the measure income they would be able to know how holidays affect their income level, how the location of hotel has affected the income time wise, etc. With such analysis they would be able to invest more in areas where income earned is higher. (Building a new hotel in a specific location, planning packages or marketing campaigns for the holidays which are profitable)

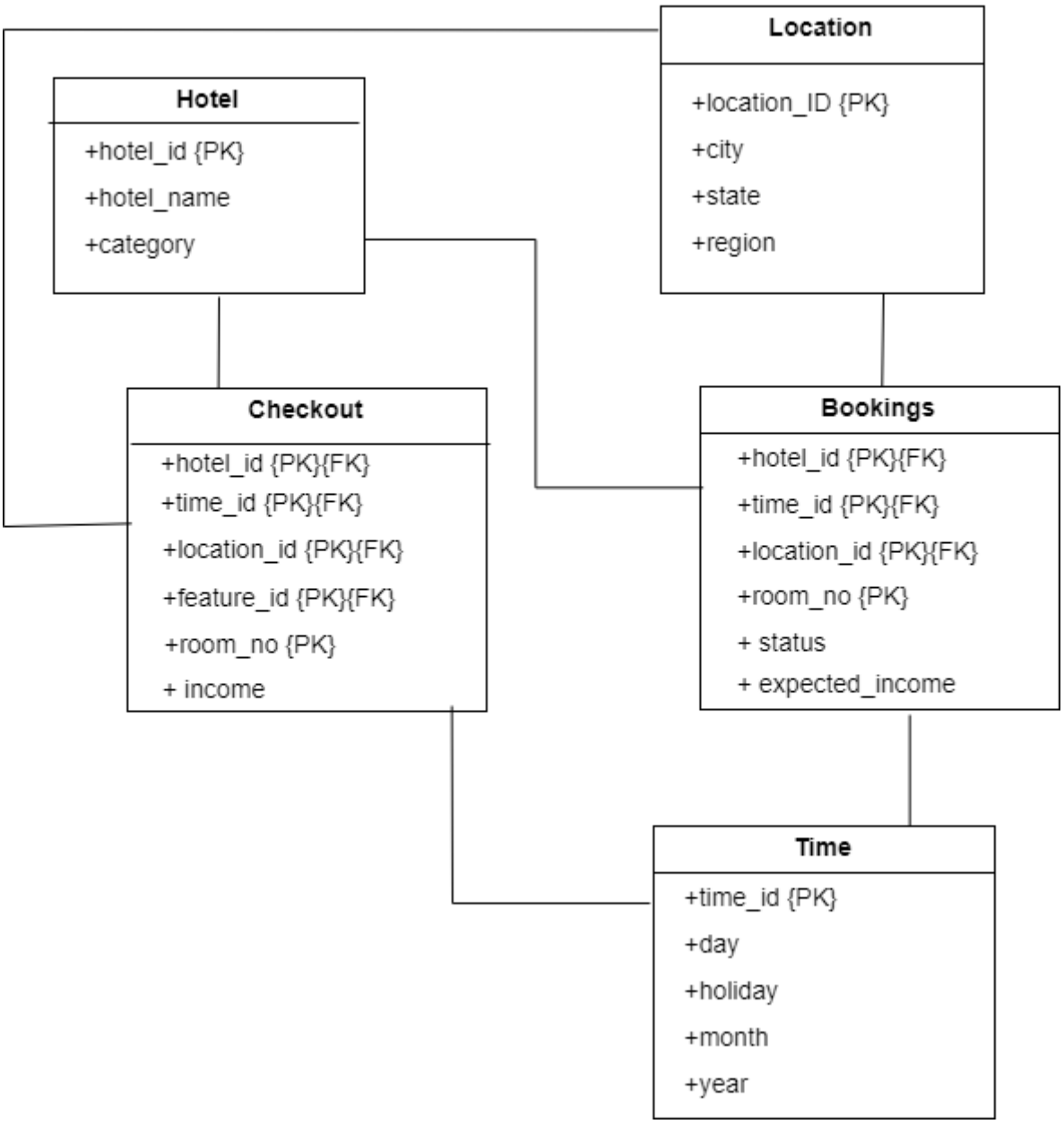
## d) IDENTIFICATION OF CONFORMING DIMENSIONS AND THEIR TREATMENT AS PART OF A SINGLE FACT SCHEMA

Hotel

Location

Time

## SINGLE FACT SCHEMA DIAGRAMS

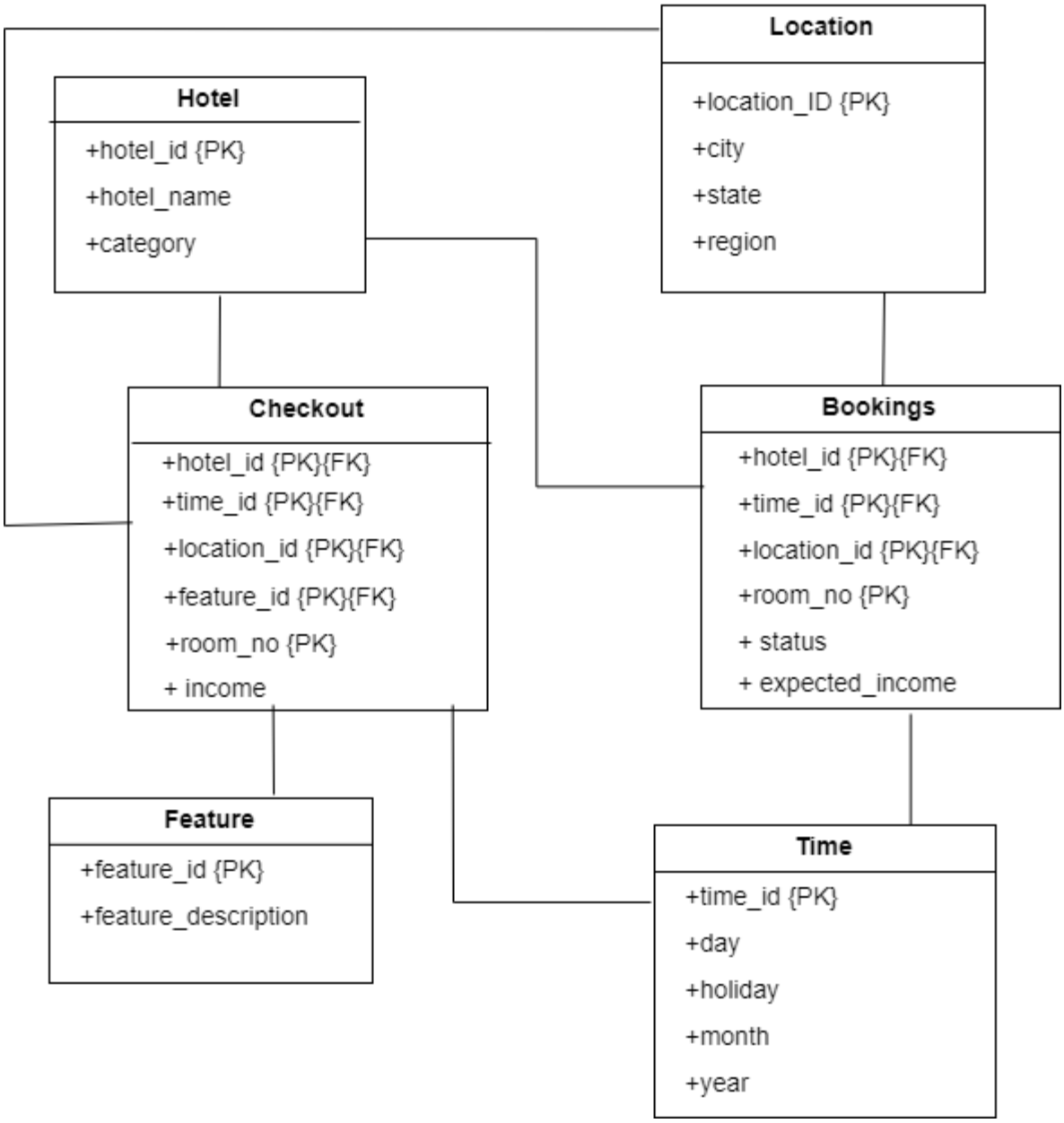



## JUSTIFICATION OF CONFORMING DIMENSIONS

Hotel, Location, Time were considered as conforming dimensions as these dimensions are common to both the facts. Feature is not a conforming dimension as it's impractical to analyze the percentages of free, reserved and unavailable rooms with the features provided for each room and make an informed decision. Moreover, analyzing the income that will be earned through checkout is more practical. Room was not considered as a conforming dimension as room dimension will be a degenerate dimension as

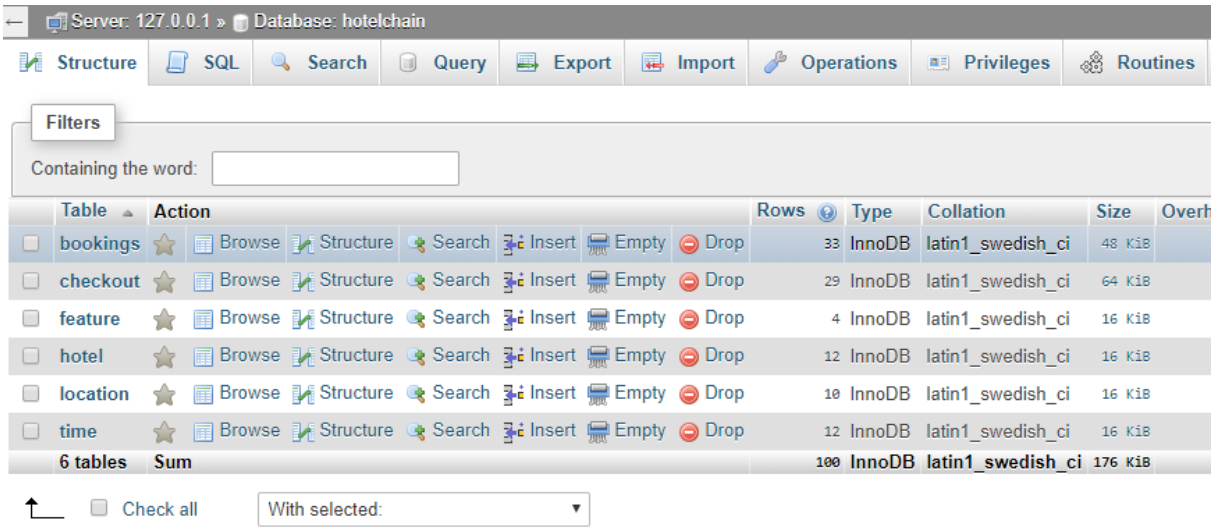
according to the source schema there's only one attribute which could be entered. Hence, primary keys were added respectively for both the fact tables.

## e) FINAL SINGLE FACT SCHEMA DIAGRAM



## f) IMPLEMENTATION OF SINGLE FACT SCHEMA USING R and MYSQL

HOTELCHAIN DATABASE IN MYSQL USING R STUDIO




### BOOKINGS FACT TABLE

```
#Creating bookings facts table
dbSendQuery(mydb, "
            CREATE TABLE bookings (
            hotel_id VARCHAR(4) NOT NULL,
            location_id VARCHAR(5) NOT NULL,
            time_id INT(8) NOT NULL,
            room_no VARCHAR(8) NOT NULL,
            status VARCHAR(15) NOT NULL,
            expected_income DOUBLE NOT NULL,
            PRIMARY KEY(hotel_id, location_id, time_id, room_no),
            FOREIGN KEY(hotel_id) REFERENCES hotel(hotel_id),
            FOREIGN KEY(location_id) REFERENCES location(location_id),
            FOREIGN KEY(time_id) REFERENCES time(time_id)
            );")
```

20150781.r × | percentage × | income ×

Source on Save | Run | Sourc

```
#Inserting records to bookings
dbSendQuery(mydb, "INSERT INTO bookings
            (hotel_id, location_id, time_id, room_no, status, expected_income)
            VALUES('RAJ1',  'RAJ11',  20170525, 'R100', 'Reserved', 10000.00),
            ('RAJ1',  'RAJ11',  20170525, 'R101', 'Not Available',  10000.00),
            ('RAJ1',  'RAJ11',  20170526, 'R100', 'Free', 10000.00),
            ('RAJ2',  'RAJ12',  20170526, 'R200', 'Free', 50000.00),
            ('RAJ2',  'RAJ12',  20170525, 'R200', 'Not Available',  50000.00),
            ('RAJ2',  'RAJ12',  20170526, 'R201', 'Free', 50000.00),
            ('RAJ2',  'RAJ12',  20170526, 'R202', 'Not Available',  50000.00),
            ('RAJ2',  'RAJ12',  20170527, 'R203', 'Not Available',  50000.00),
            ('RAJ2',  'RAJ12',  20170527, 'R204', 'Not Available',  50000.00),
            ('COL1',  'COL12',  20170627, 'R301', 'Not Available',  60000.00),
            ('COL1',  'COL12',  20170627, 'R303', 'Not Available',  60000.00),
            ('COL1',  'COL12',  20170627, 'R304', 'Reserved', 60000.00),
            ('COL2',  'DEH15',  20170627, 'R400', 'Reserved', 10000.00),
            ('COL2',  'DEH15',  20170627, 'R401', 'Reserved', 10000.00),
            ('COL2',  'DEH15',  20170627, 'R500', 'Free', 100000.00),
            ('COL2',  'DEH15',  20170627, 'R501', 'Not Available',  100000.00),
            ('COL2',  'DEH15',  20170627, 'R502', 'Not Available',  100000.00),
            ('KAD4',  'KAD14',  20170525, 'R600', 'Reserved', 210000.00),
            ('KAD4',  'KAD14',  20170525, 'R601', 'Not Available',  210000.00),
            ('KAD4',  'KAD14',  20170525, 'R602', 'Not Available',  210000.00),
            ('JAW6',  'JAW16',  20170526, 'R700', 'Reserved', 115000.00),
            ('JAW6',  'JAW16',  20170526, 'R701', 'Not Available',  115000.00),
            ('JAW6',  'JAW16',  20170526, 'R702', 'Free', 115000.00),
            ('JAW6',  'JAW16',  20170526, 'R703', 'Not Available',  115000.00),
            ('JAW6',  'JAW16',  20170527, 'R702', 'Not available',  115000.00),
            ('JAW6',  'JAW16',  20170527, 'R705', 'Not Available',  115000.00),
            ('JAW6',  'JAW16',  20170627, 'R704', 'Not Available',  115000.00),
            ('KAD4',  'KAD14',  20170625, 'R600', 'Reserved', 210000.00),
            ('KAD4',  'KAD14',  20170625, 'R601', 'Not Available',  210000.00),
            ('KAD4',  'KAD14',  20170625, 'R602', 'Free', 210000.00),
            ('JAW6',  'JAW16',  20170626, 'R700', 'Not Available',  115000.00),
            ('JAW6',  'JAW16',  20170626, 'R701', 'Not Available',  115000.00),
```

Server: 127.0.0.1 » Database: hotelchain » Table: bookings

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges

| | hotel_id | location_id | time_id | room_no | status | expected_income |
|---|---|---|---|---|---|---|
| Edit Copy Delete | COL1 | COL12 | 20170627 | R301 | Not Available | 60000 |
| Edit Copy Delete | COL1 | COL12 | 20170627 | R303 | Not Available | 60000 |
| Edit Copy Delete | COL1 | COL12 | 20170627 | R304 | Reserved | 60000 |
| Edit Copy Delete | COL2 | DEH15 | 20170627 | R400 | Reserved | 10000 |
| Edit Copy Delete | COL2 | DEH15 | 20170627 | R401 | Reserved | 10000 |
| Edit Copy Delete | COL2 | DEH15 | 20170627 | R500 | Free | 100000 |
| Edit Copy Delete | COL2 | DEH15 | 20170627 | R501 | Not Available | 100000 |
| Edit Copy Delete | COL2 | DEH15 | 20170627 | R502 | Not Available | 100000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170526 | R700 | Reserved | 115000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170526 | R701 | Not Available | 115000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170526 | R702 | Free | 115000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170526 | R703 | Not Available | 115000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170527 | R702 | Not available | 115000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170527 | R705 | Not Available | 115000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170626 | R700 | Not Available | 115000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170626 | R701 | Not Available | 115000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170626 | R702 | Free | 115000 |
| Edit Copy Delete | JAW6 | JAW16 | 20170627 | R704 | Not Available | 115000 |
| Edit Copy Delete | KAD4 | KAD14 | 20170525 | R600 | Reserved | 210000 |
| Edit Copy Delete | KAD4 | KAD14 | 20170525 | R601 | Not Available | 210000 |
| Edit Copy Delete | KAD4 | KAD14 | 20170525 | R602 | Not Available | 210000 |

Console

## CHECKOUT FACTS TABLE

```r
#Create table for chekout fact table
dbSendQuery(mydb, "
            CREATE TABLE checkout (
            hotel_id VARCHAR(4) NOT NULL,
            location_id VARCHAR(5) NOT NULL,
            time_id INT(8) NOT NULL,
            feature_id INT(2) NOT NULL,
            room_no VARCHAR(8) NOT NULL,
            income DOUBLE NOT NULL,
            PRIMARY KEY(hotel_id, location_id, time_id, feature_id, room_no),
            FOREIGN KEY(hotel_id) REFERENCES hotel(hotel_id),
            FOREIGN KEY(location_id) REFERENCES location(location_id),
            FOREIGN KEY(time_id) REFERENCES time(time_id),
            FOREIGN KEY(feature_id) REFERENCES feature(feature_id)
            );")
```

Server: 127.0.0.1 » Database: hotelchain » Table: checkout

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges

Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---|---|---|---|---|---|---|---|---|
| 1 | hotel_id | varchar(4) | latin1_swedish_ci | | No | None | | | Change |
| 2 | location_id | varchar(5) | latin1_swedish_ci | | No | None | | | Change |
| 3 | time_id | int(8) | | | No | None | | | Change |
| 4 | feature_id | int(2) | | | No | None | | | Change |
| 5 | room_no | varchar(8) | latin1_swedish_ci | | No | None | | | Change |
| 6 | income | double | | | No | None | | | Change |

20150781.r | percentage | income

Source on Save | Run

```
#Insert records to checkout
dbSendQuery(mydb, "INSERT INTO checkout
            (hotel_id, location_id, time_id, feature_id, room_no, income)
            VALUES('RAJ1',  'RAJ11',  20050525, 1,  'R100', 40000),
('RAJ1',  'RAJ11',  20050526, 2,  'R102', 60000),
            ('RAJ1',  'RAJ11',  20050527, 1,  'R100', 40000.00),
            ('RAJ1',  'RAJ11',  20050526, 2,  'R101', 60000.00),
            ('RAJ1',  'RAJ11',  20050625, 1,  'R100', 40000.00),
            ('RAJ1',  'RAJ11',  20050626, 3,  'R102', 90000.00),
            ('RAJ1',  'RAJ11',  20050627, 1,  'R100', 80000.00),
            ('RAJ1',  'RAJ11',  20050626, 2,  'R102', 60000.00),
            ('RAJ2',  'RAJ12',  20050525, 1,  'R200', 40000.00),
            ('RAJ2',  'RAJ12',  20050526, 2,  'R201', 60000.00),
            ('RAJ2',  'RAJ12',  20050525, 2,  'R202', 40000.00),
            ('RAJ2',  'RAJ12',  20050526, 3,  'R203', 60000.00),
            ('RAJ2',  'RAJ12',  20050625, 1,  'R205', 40000.00),
            ('COL1',  'COL12',  20050626, 3,  'R309', 100000.00),
            ('COL1',  'COL12',  20050625, 3,  'R309', 100000.00),
            ('COL1',  'COL12',  20050626, 2,  'R310', 60000.00),
            ('KAD4',  'KAD14',  20050525, 1,  'R600', 40000.00),
            ('KAD4',  'KAD14',  20050526, 2,  'R601', 60000.00),
            ('KAD4',  'KAD14',  20050525, 1,  'R602', 40000.00),
            ('KAD4',  'KAD14',  20050526, 2,  'R603', 90000.00),
            ('KAD4',  'KAD14',  20050625, 1,  'R604', 80000.00),
            ('KAD4',  'KAD14',  20050626, 3,  'R605', 60000.00),
            ('KAD4',  'KAD14',  20050625, 3,  'R606', 60000.00),
            ('KAD4',  'KAD14',  20050626, 2,  'R701', 60000.00),
            ('JAW6',  'JAW16',  20050525, 1,  'R702', 40000.00),
            ('JAW6',  'JAW16',  20050526, 2,  'R703', 100000.00),
            ('JAW6',  'JAW16',  20050525, 1,  'R704', 65000.00),
            ('JAW6',  'JAW16',  20050526, 2,  'R705', 65000.00),
            ('JAW6',  'JAW16',  20050625, 1,  'R706', 65000.00)
            ;")

dbListTables(mydb)
```

Console

## TIME DIMENSION

Server: 127.0.0.1 » Database: hotelchain » Table: time

Browse | Structure | SQL | Search | Insert | Export | Im

Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra |
|---|---|---|---|---|---|---|---|---|
| 1 | time_id | int(8) | | | No | None | | |
| 2 | year | int(4) | | | No | None | | |
| 3 | month | varchar(10) | latin1_swedish_ci | | No | None | | |
| 4 | day | int(2) | | | No | None | | |
| 5 | holiday | tinyint(1) | | | No | 0 | | |

20150781.r | percentage | income

Source on Save

```
            PRIMARY KEY(time_id)
            );")
#Inserting records to time
dbSendQuery(mydb, "INSERT INTO time
            (time_id, year, month, day, holiday)
            VALUES( 20170525,  2017,   'May',  25,  0),
            ( 20170526,  2017,   'May',  26,  0),
            ( 20170527,  2017,   'May',  27,  1),
            ( 20170625,  2017,   'Jun',  25,  0),
            ( 20170626,  2017,   'Jun',  26,  0),
            ( 20170627,  2017,   'Jun',  27,  1),
            ( 20050525,  2005,   'May',  25,  0),
            ( 20050526,  2005,   'May',  26,  0),
            ( 20050527,  2005,   'May',  27,  1),
            ( 20050625,  2005,   'Jun',  25,  0),
            ( 20050626,  2005,   'Jun',  26,  0),
            ( 20050627,  2005,   'Jun',  27,  1);")
```

Server: 127.0.0.1 » Database: hotelchain » Table: time

Browse | Structure | SQL | Search | Insert | Export

| | time_id | year | month | day | holiday |
|---|---|---|---|---|---|
| Edit Copy Delete | 20050525 | 2005 | May | 25 | 0 |
| Edit Copy Delete | 20050526 | 2005 | May | 26 | 0 |
| Edit Copy Delete | 20050527 | 2005 | May | 27 | 1 |
| Edit Copy Delete | 20050625 | 2005 | Jun | 25 | 0 |
| Edit Copy Delete | 20050626 | 2005 | Jun | 26 | 0 |
| Edit Copy Delete | 20050627 | 2005 | Jun | 27 | 1 |
| Edit Copy Delete | 20170525 | 2017 | May | 25 | 0 |
| Edit Copy Delete | 20170526 | 2017 | May | 26 | 0 |
| Edit Copy Delete | 20170527 | 2017 | May | 27 | 1 |
| Edit Copy Delete | 20170625 | 2017 | Jun | 25 | 0 |
| Edit Copy Delete | 20170626 | 2017 | Jun | 26 | 0 |
| Edit Copy Delete | 20170627 | 2017 | Jun | 27 | 1 |

## LOCATION DIMENSION

20150781.r | percentage | income

Source on Save

```r
library(RMySQL)

mydb <- dbConnect(MySQL(),dbname='hotelchain',
                  user='root', password="", host='localhost');

#Creting table for location
dbSendQuery(mydb, "
            CREATE TABLE location (
            location_id VARCHAR(5) NOT NULL,
            city VARCHAR(50) NOT NULL,
            state VARCHAR(50) NOT NULL,
            region VARCHAR(50) NOT NULL,
            PRIMARY KEY(location_id)
            );")
```

Server: 127.0.0.1 » Database: hotelchain » Table: location

Browse | Structure | SQL | Search | Insert | Export | Import

Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra |
|---|---|---|---|---|---|---|---|---|
| 1 | location_id | varchar(5) | latin1_swedish_ci | | No | *None* | | |
| 2 | city | varchar(50) | latin1_swedish_ci | | No | *None* | | |
| 3 | state | varchar(50) | latin1_swedish_ci | | No | *None* | | |
| 4 | region | varchar(50) | latin1_swedish_ci | | No | *None* | | |

Check all *With selected:* Browse Change Drop Primary Uni

Remove from central columns

20150781.r | percentage | income

Source on Save | Run | Sou

```
#Inserting records to location

dbSendQuery(mydb, "INSERT INTO location
(location_id, city, state, region)
            VALUES('RAJ11', 'Rajagiriya', 'Colombo',  'Western'),
            ('RAJ12', 'Welikada', 'Colombo',  'Western'),
            ('COL12', 'Colombo7', 'Colombo',  'Western'),
            ('KOT14', 'Kotte',  'Colombo',  'Western'),
            ('DEH15', 'Dehiwala', 'Colombo',  'Western'),
            ('JAW16', 'Jaela',  'Gampaha',  'Western'),
            ('KAD14', 'Kadawatha',  'Gampaha',  'Western'),
            ('KAD17', 'Kaduwela', 'Gampaha',  'Western'),
            ('KAN13', 'Kandy',  'Kandy',  'Central'),
            ('ANU15', 'Anuradhapura', 'Anuradhapura', 'North Central')
            ;")
```

Server: 127.0.0.1 » Database: hotelchain » Table: location

Browse | Structure | SQL | Search | Insert | Export | Import

+ Options

| | location_id | city | state | region |
|---|---|---|---|---|
| Edit Copy Delete | ANU15 | Anuradhapura | Anuradhapura | North Central |
| Edit Copy Delete | COL12 | Colombo7 | Colombo | Western |
| Edit Copy Delete | DEH15 | Dehiwala | Colombo | Western |
| Edit Copy Delete | JAW16 | Jaela | Gampaha | Western |
| Edit Copy Delete | KAD14 | Kadawatha | Gampaha | Western |
| Edit Copy Delete | KAD17 | Kaduwela | Gampaha | Western |
| Edit Copy Delete | KAN13 | Kandy | Kandy | Central |
| Edit Copy Delete | KOT14 | Kotte | Colombo | Western |
| Edit Copy Delete | RAJ11 | Rajagiriya | Colombo | Western |
| Edit Copy Delete | RAJ12 | Welikada | Colombo | Western |

## HOTEL DIMENSION

```
#Creating table for hotel

dbSendQuery(mydb, "
            CREATE TABLE hotel (
            hotel_id VARCHAR(4) NOT NULL,
            hotel_name VARCHAR(20) NOT NULL,
            category VARCHAR(10) NOT NULL,
            PRIMARY KEY(hotel_id)
            );")
```

Server: 127.0.0.1 » Database: hotelchain » Table: hotel

Browse | Structure | SQL | Search | Insert | Export | In

Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Ex |
|---|---|---|---|---|---|---|---|---|
| 1 | hotel_id | varchar(4) | latin1_swedish_ci | | No | *None* | | |
| 2 | hotel_name | varchar(20) | latin1_swedish_ci | | No | *None* | | |
| 3 | category | varchar(10) | latin1_swedish_ci | | No | *None* | | |

```
#Inserting records to hotel
dbSendQuery(mydb, "INSERT INTO hotel
(hotel_id, hotel_name, category)
            VALUES('RAJ1',  'Topaz',  '4 Stars'),
            ('RAJ2',  'Senora', '4 Stars'),
            ('RAJ3',  'LemonGrass', '5 Stars'),
            ('RAJ4',  'Siyenra',  '3 Stars'),
            ('RAJ5',  'Gornea', '4 Stars'),
            ('COL1',  'Cinamon Red',  '5 Stars'),
            ('COL2',  'Java Lounge',  '4 Stars'),
            ('COL3',  'Cinamon Citadel',  '3 Stars'),
            ('COL5',  'Cinamon Anurapurs',  '4 Stars'),
            ('COL6',  'Cinamon Blue', '4 Stars'),
            ('KAD4',  'Cinamon Kadawatha',  '4 Stars'),
            ('JAW6',  'cinew',  '4 Stars');")
```

Server: 127.0.0.1 » Database: hotelchain » Table: hotel

Browse | Structure | SQL | Search | Insert | Ex

+ Options

| | hotel_id | hotel_name | category |
|---|---|---|---|
| Edit Copy Delete | COL1 | Cinamon Red | 5 Stars |
| Edit Copy Delete | COL2 | Java Lounge | 4 Stars |
| Edit Copy Delete | COL3 | Cinamon Citadel | 3 Stars |
| Edit Copy Delete | COL5 | Cinamon Anurapurs | 4 Stars |
| Edit Copy Delete | COL6 | Cinamon Blue | 4 Stars |
| Edit Copy Delete | JAW6 | cinew | 4 Stars |
| Edit Copy Delete | KAD4 | Cinamon Kadawatha | 4 Stars |
| Edit Copy Delete | RAJ1 | Topaz | 4 Stars |
| Edit Copy Delete | RAJ2 | Senora | 4 Stars |
| Edit Copy Delete | RAJ3 | LemonGrass | 5 Stars |
| Edit Copy Delete | RAJ4 | Siyenra | 3 Stars |
| Edit Copy Delete | RAJ5 | Gornea | 4 Stars |

## FEATURE DIMENSION

20150781.r | percentage | income

Source on Save

```
#Create table for feature
dbSendQuery(mydb, "
            CREATE TABLE feature (
            feature_id INT(2) NOT NULL,
            feature_description VARCHAR(80) NOT NULL,
            PRIMARY KEY(feature_id)
            );")
```

Server: 127.0.0.1 » Database: hotelchain » Table: feature

Browse | Structure | SQL | Search | Insert | Export

Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comr |
|---|---|---|---|---|---|---|---|
| 1 | feature_id | int(2) | | | No | *None* | |
| 2 | feature_description | varchar(80) | latin1_swedish_ci | | No | *None* | |

20150781.r | percentage | income

Source on Save

```
#Inserting records to feature
dbSendQuery(mydb, "INSERT INTO feature
            (feature_id, feature_description)
            VALUES(1, '1 Bed, TV'),
            (2, '2 beds, TV'),
            (3, '3 Beds, TV, Fridge'),
            (4, '4 beds, TV, Fridge, Watertub');")
```

Server: 127.0.0.1 » Database: hotelchain » Table: feature

Browse | Structure | SQL | Search | Insert | Ex

Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

| | feature_id | feature_description |
|---|---|---|
| Edit Copy Delete | 1 | 1 Bed, TV |
| Edit Copy Delete | 2 | 2 beds, TV |
| Edit Copy Delete | 3 | 3 Beds, TV, Fridge |
| Edit Copy Delete | 4 | 4 beds, TV, Fridge, Watertub |

Check all | With selected: Edit | Copy | Delete | E

## 2. OLAP QUERIES

a) In 2017, for each state and month, analyze the portion of rooms which are reserved, free and unavailable?

- **Query Result**

20150781.r* percentage income

Filter

| | YEAR | MONTH | STATE | PERCENTAGE OF FREE ROOMS | PERCENTAGE OF RESERVED ROOMS | PERCENTAGE OF UNAVAILABLE ROOMS |
|---|---|---|---|---|---|---|
| 1 | 2017 | Jun | Colombo | 12.5000 | 37.5000 | 50.0000 |
| 2 | 2017 | Jun | Gampaha | 28.5714 | 14.2857 | 57.1429 |
| 3 | 2017 | May | Colombo | 33.3333 | 11.1111 | 55.5556 |
| 4 | 2017 | May | Gampaha | 11.1111 | 22.2222 | 66.6667 |

- **SELECT Query to calculate the percentages of free rooms, reserved rooms and unavailable rooms**

20150781.r* × percentage × income ×

Source on Save Run Source

```
#calculating the portion of reserved rooms, free rooms and unavailable rooms

percentage = fetch(dbSendQuery(mydb, "SELECT t.year AS YEAR, t.month AS MONTH, l.state AS STATE,

      (COUNT(CASE WHEN b.status='Free' THEN 1 END)/COUNT(CASE WHEN b.status='Free' OR
                b.status = 'Not Available' or
                b.status = 'Reserved' THEN 1 END)*100)
      AS 'PERCENTAGE OF FREE ROOMS',


      (COUNT(CASE WHEN b.status='Reserved' THEN 1 END)/COUNT(CASE WHEN b.status='Free' OR
                b.status = 'Not Available' or
                b.status = 'Reserved' THEN 1 END)*100)
      AS 'PERCENTAGE OF RESERVED ROOMS',

      (COUNT(CASE WHEN b.status='Not Available' THEN 1 END)/COUNT(CASE WHEN b.status='Free' OR
                b.status = 'Not Available' or
                              b.status = 'Reserved' THEN 1 END)*100)
                              AS 'PERCENTAGE OF UNAVAILABLE ROOMS'  FROM bookings b, time t, location l

                WHERE b.time_id = t.time_id AND b.location_id = l.location_id AND
      t.year= 2017 GROUP BY MONTH, STATE;"))


head(percentage)
```

b) In 2005, for each state and month, analyze the income of 4 star hotels and the cumulative income of 4 star hotels?

## INCOME ANALYSIS

```
#analyzing the income of 4-star hotels for the given criteria in question

income = fetch(dbSendQuery(mydb, "SELECT time.year AS YEAR,
                          time.month AS MONTH,
                          location.state AS STATE,
                          SUM(checkout.income) AS 'INCOME OF 4 STAR HOTELS'
                          FROM hotel, time, location, checkout
                          WHERE time.year = 2005 AND hotel.category = '4 Stars' AND
                          time.time_id=checkout.time_id AND
                          hotel.hotel_id=checkout.hotel_id AND
                          location.location_id=checkout.location_id
                          GROUP BY MONTH, STATE ORDER BY YEAR;"))
head(income)
```

| | YEAR | MONTH | STATE | INCOME OF 4 STAR HOTELS |
|---|---|---|---|---|
| 1 | 2005 | May | Gampaha | 500000 |
| 2 | 2005 | Jun | Gampaha | 325000 |
| 3 | 2005 | May | Colombo | 400000 |
| 4 | 2005 | Jun | Colombo | 310000 |

## CUMULATIVE INCOME FROM THE ABOVE ANALYSIS

*There are 2 methods which has being used to calculate the total cumulative income

20150781.r | percentage | income

Source on Save | Run | Source

```
                                    hotel.hotel_id=checkout.hotel_id AND
                                    location.location_id=checkout.location_id
                                    GROUP BY MONTH, STATE ORDER BY YEAR;"))
head(income)

#Calculating the total cumulative income year of all 4 Star Hotels for the given criteria in question

#1 st method - without adding a column retreiving the final cumulative income

TotalCumulativeincome_4StarHotels <- sum(income$INCOME)
print(TotalCumulativeincome_4StarHotels)


#second method -  by adding a column

Cumulativeincome_4StarHotels <- cumsum(income$INCOME)
income[, "CUMULATIVE INCOME OF 4 STAR HOTELS"] <- Cumulativeincome_4StarHotels <- cumsum(income$INCOME)
head(income)

#Clearing data
rm(Cumulativeincome_4StarHotels)
rm(TotalCumulativeincome_4StarHotels)
```

## First Method Result

```
                                hotel.hotel_id=checkout.hotel_id AND
                                location.location_id=checkout.location_id
                                GROUP BY MONTH, STATE ORDER BY YEAR;"))
head(income)

#Calculating the total cumulative income year of all 4 Star Hotels for the given criteria in question

#1st method - without adding a column retreiving the final cumulative income

TotalCumulativeincome_4StarHotels <- sum(income$INCOME)
print(TotalCumulativeincome_4StarHotels)


#second method -  by adding a column

Cumulativeincome_4StarHotels <- cumsum(income$INCOME)
income[, "CUMULATIVE INCOME OF 4 STAR HOTELS"] <- Cumulativeincome_4StarHotels <- cumsum(income$INCOME)
head(income)

#Clearing data
rm(Cumulativeincome_4StarHotels)
rm(TotalCumulativeincome_4StarHotels)
```

```
> TotalCumulativeincome_4StarHotels <- sum(income$INCOME)
> print(TotalCumulativeincome_4StarHotels)
[1] 1535000
```

## Second Method Result

20150781.r* × percentage × income ×

Filter

| | YEAR | MONTH | STATE | INCOME OF 4 STAR HOTELS | CUMULATIVE INCOME OF 4 STAR HOTELS |
|---|---|---|---|---|---|
| 1 | 2005 | May | Gampaha | 500000 | 500000 |
| 2 | 2005 | Jun | Gampaha | 325000 | 825000 |
| 3 | 2005 | May | Colombo | 400000 | 1225000 |
| 4 | 2005 | Jun | Colombo | 310000 | 1535000 |

## 3. SUITABILITY OF R TO PERFORM BI QUERIES ON BIG DATA REPOSITORIES

With Digitalization and rapid growth of technologies, increasing number of organizations are experiencing difficulties with explosion of data and the sizes of databases used are growing at exponential rates. Thus, with heterogeneous data generated through daily operations it is a challenging task to process and analyze the data in order to extract meaningful information. Hence, with time it's becoming difficult to for the business to manage by analyzing data with heavy volume, volatility and variety. Due to such characteristics of data it is challenging to analyze data with traditional means such as analyzing through relational database management systems and desktop software packages for statistics and visualization. (Bogdon, 2016) At present, Hadoop framework which consists of libraries, , distributed file system (HDFS), and resource management platform and implements a version of the MapReduce programming model for processing large-scale data is widely used for storage and processing of big data clusters of community hardware. Therefore, in this research paper different methods and approaches used with R is discussed as follows.

### 1) INTEGRATION OF DATA USING R AND HADOOP

According to Uskenbayeva, et al.(2015), have identified that it is ideal to use Apache Hadoop and programming language R which can ensure the integrity of data during the integration.

There are mainly 3 approaches to integrate R with Hadoop namely;

- R and streaming

Integrating R and Hadoop using Streaming is an easy task because the user only has to run Hadoop command line to launch Streaming job as command line arguments. (Bogdan, 2014) By using streaming together with R scripts in the map/reduce space since R can read or write data from/to standard point. Moreover, there is no client-side integration with R. (Uskenbayeva,2015) However, R should be installed on every DataNode of the Hadoop cluster.

- Rhipe

Rhipe provides a tight integration between R and Hadoop which stands for "R and Hadoop Integrated Programming Environment" which allows users to analyse big-data directly in R. Moreover, it is comparatively a difficult task as for each DataNode the user should install R, Protocol, Buffers and Rhipe and also R database should be shared library. However, benefits provide a defile system across a cluster of computers which optimizes the processor usage, fault tolerance. It allows users to call and map reduce functions within R itself.

- RHadoop

RHadoop provides client-site integration of R and Hadoop. Setting up RHadoop is fairly easy task. Thus, there are dependencies on each data node of Data Cluster which needs to be considered.

Therefore, the above-mentioned approaches have advantages and disadvantages as follow. Using R and Streaming has no disturbance regarding installation, where Rhipe and Rhadoop require some effort in order to set up the cluster. Integrating with R from the client-site part is high for Rhipe and Rhadoop and less for R and Streaming. As Rhipe and Rhadoop are open source projects the software packages are freely available for download. Rhipe and Rhadoop allows users to define their own map and reduce functions within R, while streaming uses a command line approach to define them.

## 2) INTEGRATION OF DATA USING R AND ORACLE

Oracle has adopted R as a language and environment to support performing statistical data analysis, advanced analysis and generating graphics. Oracle R Distribution, Oracle R Enterprise, Oracle R Advanced and ROracle are the four key technologies which provides integration with R language. Using Oracle databases for big-data analysis has many benefits including, eliminates data movement to analytic servers, allows ability to analyse entire data, ability to use database infrastructure in a greater way, provides good scalability and speed and control CRAN algorithms clearly in the databases. There are some limitations when integrating oracle and R. Complex engines and it requires skills for the installation are some of them. Moreover, oracle takes very long time to develop, is not answering instantly and it’s complicated ETL are some limitations in using oracle in terms of business process.

## 3) INTEGRATION OF DATA USING R AND TERADATA

Teradata is a fully accessible relational database management system produced by Teradata Corp. in 2013 Teradata has introduced its new database, database 14.10 which easily gives access to R programming language. With this integration, R users will soon be able to use the power of the Teradata Database as a massively-parallel R platform and use the parallel-external memory algorithms of Revolution R Enterprise ScaleR for advanced data processing and statistical modelling with big data. This has many positive results such as, computational parallelism that dramatically accelerates the delivery of results from data, which eliminates the need to move data to a middle tier for analysis,

eliminates the need to move data to analytics servers before initiating exploration, modelling or scoring, enabling developers to run more sophisticated and more numerous analytic models and reduces dependence on IT staff to move data, freeing developers to pursue more forward-thinking projects etc. Teradata is not capable of sharing and network BYNET V5 has scalability up to 2048 nodes, 76Tb(10K) per node, in total 324Pb, poor producing language and tools are some limitations in this. (Calif , 2013)

Therefore, in conclusion by observing the above benefits and limitation it was realized that, Oracle could provide a good integrating with R, which has lots of advantages against others. It provides most rich tools and fast development. Teradata also has powerful SQL languages but with few storages and poor produce languages and tools. In addition, Oracle has 3 caches such as memory on Storage Cell, SDD PCI Flash cards and memory of main server, where Teradata provides one cache memory of nodes which knows the temperature of storing. Hadoop can be very fast, but only in very specific tasks in a specific way. Teradata provides fastest development in specific ways, but user cannot make changes of process and even though oracle is slower on big-data it allows access to many databases which has thousands of features and optimizations. Finally, it was considered that Oracle can provide best features with R integration comparing to other technologies.

## REFERENCES

Uskenbayeva, R., Kuandykov, a., Cho, Y., Temirbolatova, T., Amanzholova, S. and Kozhamzharova, D. (2015). Integrating of Data Using the Hadoop and R. Procedia Computer Science, 56, pp.145-149.

Oracle.com. (2017). Oracle R Technologies. [online] Available at: http://www.oracle.com/technetwork/topics/bigdata/r-offerings-1566363.html [Accessed 16 Nov. 2018].

OANCEA, B. (2014). Integrating R and Hadoop for Big Data Analysis. *Revista română de statistică: organ al Comisiei Naționale pentru Statistică*. [online] Available at: https://www.researchgate.net/publication/262378989_Integrating_R_and_Hadoop_for_Big_Data_Analysis [Accessed 16 Nov. 2018].

Teradata.com. (2017). Teradata Offers First, Fully-Parallel, Scalable R Analytics. [online] Available at: http://www.teradata.com/Press-Releases/2013/Teradata-Offers-First,-Fully-Parallel,-Scalab [Accessed 16 Nov. 2018].

Smith, D. (2013). Revolution Analytics and Teradata bring R into the Database. [online] R-bloggers. Available at: https://www.r-bloggers.com/revolution-analytics-and-teradata-bring-r-into-the-database/ [Accessed 16 Nov. 2018].

Techopedia.com. (2017). What is Teradata? - Definition from Techopedia. [online] Available at: https://www.techopedia.com/definition/25987/teradata [Accessed 16 Nov. 2018].

Calif , S. D., 2013. Teradata Offers First, Fully-Parallel, Scalable R Analytics. *teradata.com.*

Uskenbayeva, R. et al., 2015. Integrating of data using the Hadoop and R. *The 12th International Conference on Mobile Systems and Pervasive Computing ,* p. 145 – 149 .