



# Vidyavardhini's College of Engineering & Technology

Department of Artificial Intelligence and Data Science

---

<b>Name:</b>	
<b>Roll No:</b>	
<b>Class/Sem:</b>	TE/V
<b>Experiment No.:</b>	2
<b>Title:</b>	Implementation of Dimension tables and Fact tables and perform OLAP operations.
<b>Date of Performance:</b>	
<b>Date of Submission:</b>	
<b>Marks:</b>	
<b>Sign of Faculty:</b>	



**Aim:** Implementation of Dimension and Fact tables and perform OLAP operations.

**Objective:** OLAP stands for Online Analytical Processing. The objective of OLAP is to analyze information from multiple database systems at the same time. It is based on multidimensional data model and allows the user to query on multi-dimensional data.

### Theory:

- Online Analytical Processing Server (OLAP) is based on the multidimensional data model.
- The main aim of OLAP is to provide multidimensional analysis to the underlying data. Following is the list of OLAP operations:
  1. Roll-up
  2. Drill-down
  3. Slice
  4. Dice
  5. Pivot (rotate)

### Roll-up:

- The roll-up operation (also called the drill-up operation) performs aggregation on a data cube, either by climbing up a concept hierarchy for a dimension or by dimension reduction.
- Figure 2.1 shows the result of a roll-up operation performed on the central cube by climbing up the concept hierarchy for location.
- This hierarchy was defined as the total order “street < city < province or state < country.”
- The roll-up operation aggregates the data by ascending the location hierarchy from the level of city to the level of country.
- In other words, rather than grouping the data by city, the resulting cube groups the data by country.

### Drill-down:

- Drill-down is the reverse of roll-up. It navigates from less detailed data to more detailed data.



# Vidyavardhini's College of Engineering & Technology

## Department of Artificial Intelligence and Data Science

---

- Drill-down can be realized by either stepping down a concept hierarchy for a dimension or introducing additional dimensions.
- Figure 2.1 shows the result of a drill-down operation performed on the central cube by stepping down a concept hierarchy for time defined as “day < month < quarter < year.”
- Drill-down occurs by descending the time hierarchy from the level of quarter to the more detailed level of month.
- The resulting data cube details the total sales per month rather than summarizing them by quarter.

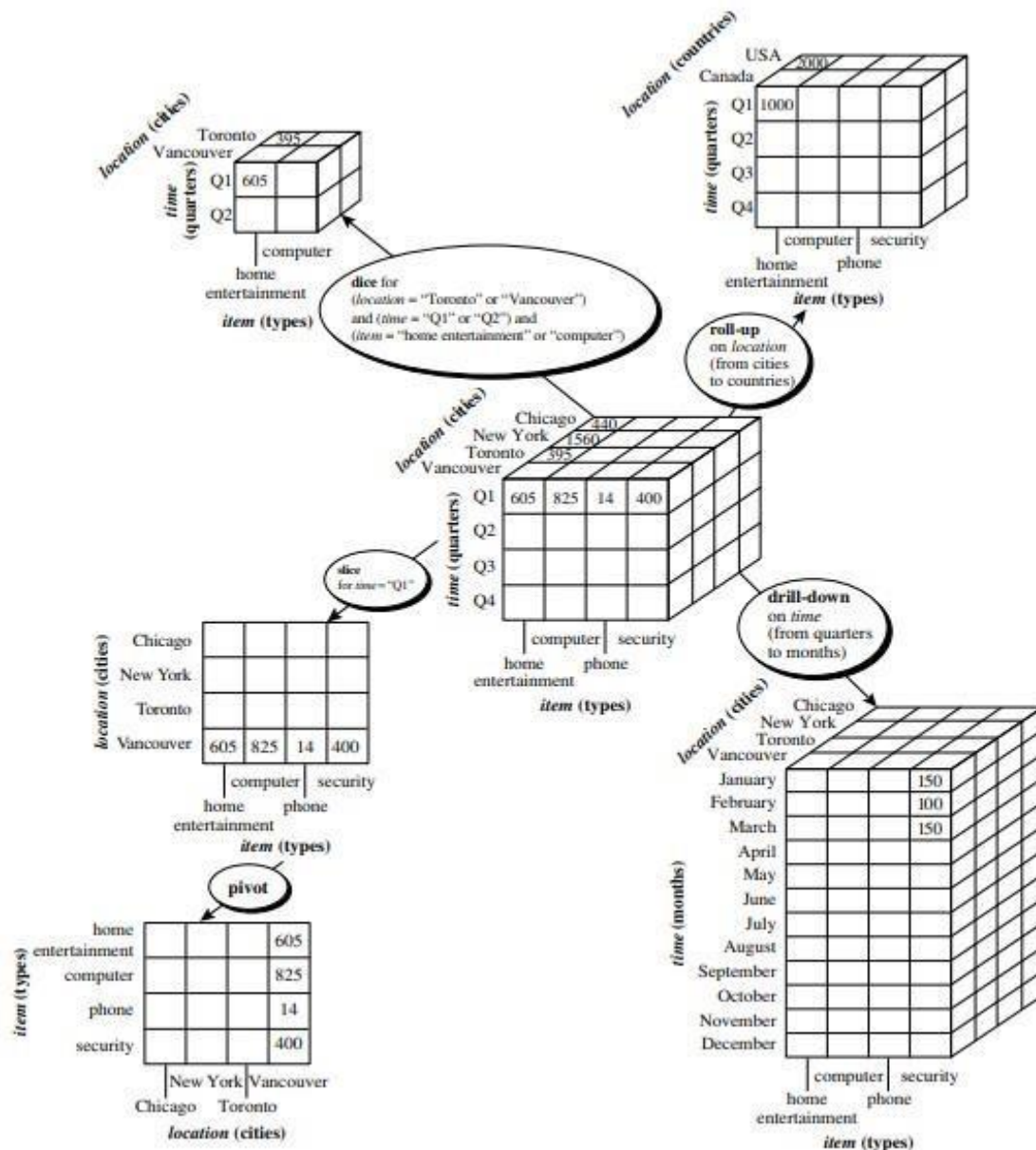


Figure 2.1: Examples of typical OLAP operations on multidimensional data.

### Slice:

- The slice operation performs a selection on one dimension of the given cube, resulting in a subcube.
- Figure 2.1 below shows a slice operation where the sales data are selected from the central cube for the dimension time using the criterion time = "Q1."



### **Dice:**

- The dice operation defines a subcube by performing a selection on two or more dimensions.
- Figure 2.1 shows a dice operation on the central cube based on the following selection criteria that involve three dimensions: (location = "Toronto" or "Vancouver") and (time = "Q1" or "Q2") and (item = "home entertainment" or "computer").

### **Pivot:**

- Pivot (also called rotate) is a visualization operation that rotates the data axes in view to provide an alternative data presentation.
- Figure 2.1 shows a pivot operation where the item and location axes in a 2-D slice are rotated.

### **Problem Statement:**

Suppose that a data warehouse consists of four dimensions as patient, doctor, location and treatment. The two measures are count and fees, where fees is the treatment charge paid by the patient to the doctor on a weekly basis. Draw a star and snowflake schema diagram for the above data warehouse.

### **Output:**

1. Creating the Dimension Tables

Create Database Hospital;

```
CREATE TABLE patient(  
    Patient_id int(10) PRIMARY KEY,  
    Patient_name varchar(50),  
    Patient_age int(10),  
    Patient_address varchar(250),  
    Report_id int(10)  
);
```

```
CREATE TABLE doctor(  
    Doctor_id int(10) PRIMARY KEY,
```



```
Doctor_name varchar(50),  
Doctor_type varchar(50),  
Doctor_age int(10),  
Doctor_experience varchar(250),  
1st_week int(10),  
2nd_week int(10),  
3rd_week int(10),  
4th_week int(10)  
);
```

```
CREATE TABLE Location (  
    location_id int(10) PRIMARY KEY,  
    city varchar(50), state varchar(50),  
    pincode int(10), country varchar(50)  
);
```

```
create table treatment(  
    Treatment_id int(10) PRIMARY KEY,  
    Treatment_name varchar(50),  
    Treatment_duration varchar(20),  
    Treatment_type varchar(20)  
);
```

## 2. Creating the Fact Table

```
CREATE TABLE fact_table(  
    Doctor_id int(10) REFERENCES doctor(Doctor_id), Patient_id  
    int(10) REFERENCES patient(Patient_id), location_id int(10)
```



# Vidyavardhini's College of Engineering & Technology

## Department of Artificial Intelligence and Data Science

---

REFERENCES location(location\_id), treatment\_id int(10)

REFERENCES treatment(Treatment\_id),

Count int(10),

Fees int(10)

);

### 3. Inserting values in both dimension and fact tables

INSERT INTO location(location\_id,city,state,pincode,country)

VALUES(1234567789,'vasai','maharashtra',301303,'India'),

(1234567779,'vasai','maharashtra',401202,'India'),

(1234567079,'vasai','maharashtra',401206,'India'),

(1234567059,'vasai','maharashtra',401106,'India');

INSERT INTO

treatment(Treatment\_id,Treatment\_name,Treatment\_duration,Treatment\_type)

VALUES(101,"Angioplasty","3 Hours","Heart Disease"),

(102,"Chemotherapy","8 Hours","Lungs Disease"),

(103,"Heart Bypass Surgery","14 Hours","Heart Disease"), (104,"Lead Extraction","12  
Hours","Heart Disease"),

(105,"Dialysis","5 Hours","Kidney Disease");

INSERT INTO fact\_table(Doctor\_id,Patient\_id,location\_id,treatment\_id,Count,Fees)

VALUES (01,62,1234567789,101,31,500),

(02,66,1234567779,102,32,700),

(01,60,1234567079,103,33,800),

(02,61,1234567059,104,34,900);



INSERT INTO

doctor(Doctor\_id,Doctor\_name,Doctor\_type,Doctor\_age,Doctor\_experience,1st\_week,2nd\_week,3rd\_week,4th\_week)

VALUES (01,'Sairaj','Brain',20,'1 years',100,200,100,100),

(02,'Viraj','Kidney',20,'5 years',200,100,100,300),

(03,'Shivansh','Heart',21,'7 years',200,200,200,200),

(04,'Anand','Lung',19,'4 years',300,300,200,100); 4.

Displaying the tables Location:

Showing rows 0 - 3 (4 total, Query took 0.0002 seconds.)

SELECT \* FROM `location`

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

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

Extra options

	location_id	city	state	pincode	country
<input type="checkbox"/> Edit Copy Delete	1234567059	vasai	maharashtra	401106	India
<input type="checkbox"/> Edit Copy Delete	1234567079	vasai	maharashtra	401206	India
<input type="checkbox"/> Edit Copy Delete	1234567779	vasai	maharashtra	401202	India
<input type="checkbox"/> Edit Copy Delete	1234567789	vasai	maharashtra	301303	India

☐ Check all | With selected: Edit Copy Delete Export

Patient:





# Vidyavardhini's College of Engineering & Technology

## Department of Artificial Intelligence and Data Science

The screenshot shows the MySQL Workbench interface. On the left, the 'Schemas' pane lists various databases, including 'hospitals', 'information\_schema', 'mysql', 'performance\_schema', 'phpmyadmin', 'sales\_analysis', and 'test'. The 'hospitals' database is selected, and the 'patient' table is highlighted. The main window displays the 'SELECT \* FROM `patient`' query. Below the query, there are options for 'Profiling', 'Edit inline', 'Edit', 'Explain SQL', 'Create PHP code', and 'Refresh'. A 'Show all' checkbox is present, along with a 'Number of rows' dropdown set to 25, a 'Filter rows' search box, and a 'Sort by key' dropdown set to 'None'. An 'Extra options' button is also visible. The table data is displayed in a grid with columns: Patient\_id, Patient\_name, Patient\_age, Patient\_address, and Report\_id. The data rows are:

	Patient_id	Patient_name	Patient_age	Patient_address	Report_id
<input type="checkbox"/>	60	Shivansh	20	Vasai	4
<input type="checkbox"/>	61	Anand	20	Vasai	5
<input type="checkbox"/>	62	Sairaj	20	Virar	2
<input type="checkbox"/>	66	Viraj	20	Goregaon	3

Doctor:

The screenshot shows the MySQL Workbench interface. The top toolbar includes buttons for 'Browse', 'Structure', 'SQL', 'Search', 'Insert', 'Export', 'Import', 'Privileges', 'Operations', 'Tracking', and 'More'. The main window displays the 'SELECT \* FROM `doctor`' query. Below the query, there are options for 'Profiling', 'Edit inline', 'Edit', 'Explain SQL', 'Create PHP code', and 'Refresh'. A 'Show all' checkbox is present, along with a 'Number of rows' dropdown set to 25, a 'Filter rows' search box, and a 'Sort by key' dropdown set to 'None'. An 'Extra options' button is also visible. The table data is displayed in a grid with columns: Doctor\_id, Doctor\_name, Doctor\_type, Doctor\_age, Doctor\_experience, 1st\_week, 2nd\_week, 3rd\_week, and 4th\_week. The data rows are:

	Doctor_id	Doctor_name	Doctor_type	Doctor_age	Doctor_experience	1st_week	2nd_week	3rd_week	4th_week
<input type="checkbox"/>	1	Sairaj	Brain	20	1 years	100	200	100	100
<input type="checkbox"/>	2	Viraj	Kidney	20	5 years	200	100	100	300
<input type="checkbox"/>	3	Shivansh	Heart	21	7 years	200	200	200	200
<input type="checkbox"/>	4	Anand	Lung	19	4 years	300	300	200	100

Treatment:



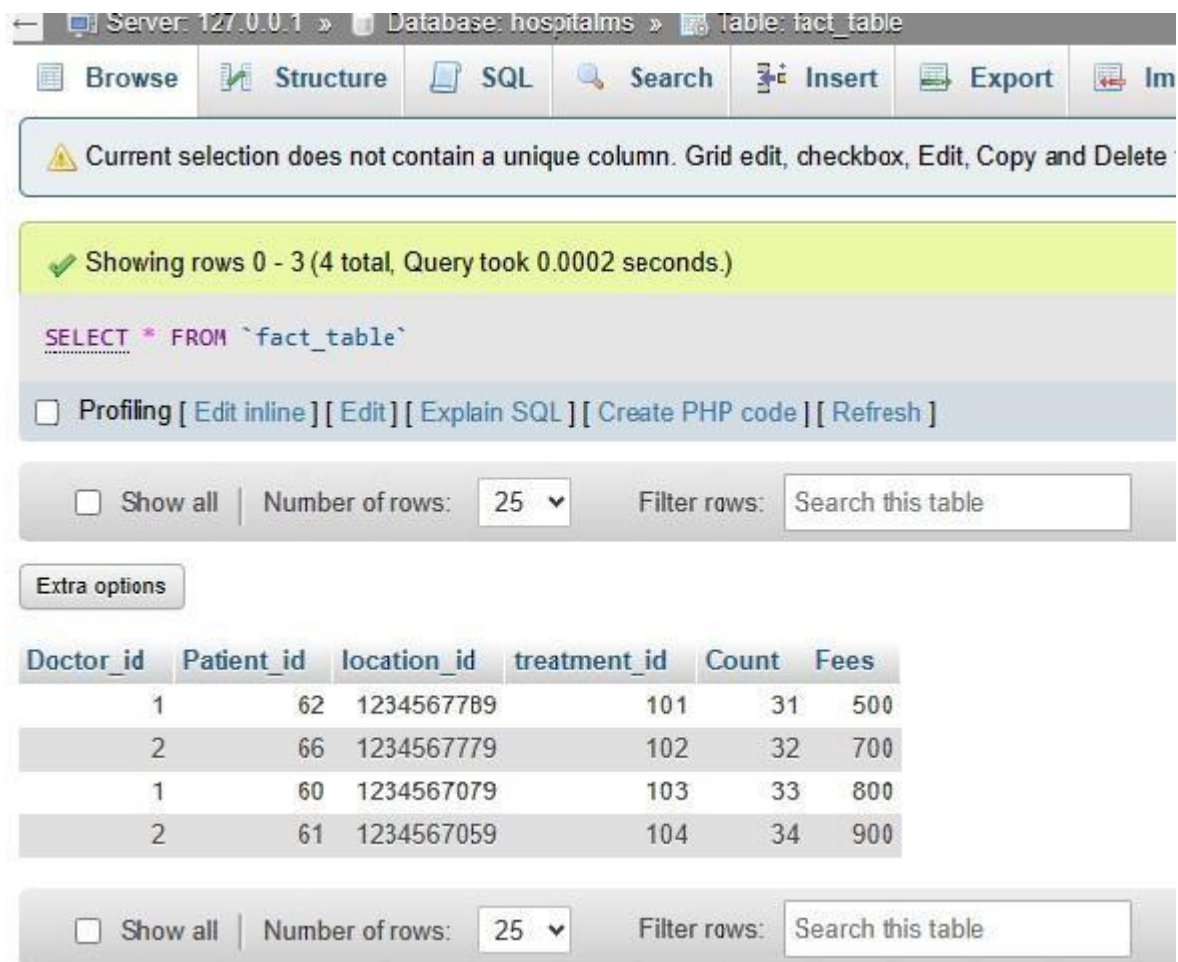
# Vidyavardhini's College of Engineering & Technology

Department of Artificial Intelligence and Data Science



Treatment_id	Treatment_name	Treatment_duration	Treatment_type
101	Angioplasty	3 Hours	Heart Disease
102	Chemotherapy	8 Hours	Lungs Disease
103	Heart Bypass Surgery	14 Hours	Heart Disease
104	Lead Extraction	12 Hours	Heart Disease
105	Dialysis	5 Hours	Kidney Disease

Fact Table:



Doctor_id	Patient_id	location_id	treatment_id	Count	Fees
1	62	1234567789	101	31	500
2	66	1234567779	102	32	700
1	60	1234567079	103	33	800
2	61	1234567059	104	34	900



5. Write SQL Queries for all the above OLAP operations.

Roll Up :

```
SELECT d.Doctor_id, d.Doctor_name, d.Doctor_type, SUM(f.Fees) AS Total_fees
```

```
From doctor AS d
```

```
JOIN fact_table AS f
```

```
Where d.Doctor_id = f.Doctor_id
```

```
Group BY d.Doctor_id ;
```

The screenshot displays a database management interface with a toolbar at the top containing icons for Browse, Structure, SQL, Search, Insert, Export, and Import. Below the toolbar is a 'Show query box' button. A warning message states: 'Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete functions are disabled.' Below this, a green status bar indicates 'Showing rows 0 - 1 (2 total, Query took 0.0003 seconds.)'. The SQL query is displayed in a text area: 

```
SELECT d.Doctor_id, d.Doctor_name, d.Doctor_type, SUM(f.Fees) AS Total_fees From doctor AS d JOIN fact_table AS f ON d.Doctor_id = f.Doctor_id Group BY d.Doctor_id;
```

 Below the query area are links for Profiling, Edit inline, Edit, Explain SQL, Create PHP code, and Refresh. A control bar shows 'Show all', 'Number of rows: 25', and a 'Filter rows: Search this table' input. An 'Extra options' button is also present. The query results are shown in a table with columns Doctor\_id, Doctor\_name, Doctor\_type, and Total\_fees. The results are as follows:

Doctor_id	Doctor_name	Doctor_type	Total_fees
1	Sairaj	Brain	1300
2	Viraj	Kidney	1600

At the bottom, there is another control bar with 'Show all', 'Number of rows: 25', and a 'Filter rows: Search this table' input.

Drill Down :



Select d.Doctor\_name, d.1st\_week,d.2nd\_week,3rd\_week,4th\_week,f.Fees

FROM doctor as d

JOIN fact\_table AS f Where d.Doctor\_id = f.Doctor\_id

Group BY d.Doctor\_name ;

Show query box

⚠ Current selection does not contain a unique column. Grid edit, checkbox, E

✓ Showing rows 0 - 1 (2 total, Query took 0.0002 seconds.)

```
Select d.Doctor_name, d.1st_week,d.2nd_week,3rd_week,4th_week,f.  
Group BY d.Doctor_name;
```

☐ Profiling [ [Edit inline](#) ] [ [Edit](#) ] [ [Explain SQL](#) ] [ [Create PHP code](#) ] [ [Refresh](#) ]

☐ Show all | Number of rows: 25 ▼ Filter rows: Search this t

Extra options

Doctor_name	1st_week	2nd_week	3rd_week	4th_week	Fees
Sairaj	100	200	100	100	500
Viraj	200	100	100	300	700

Slice :

Select t.Treatment\_name,t.Treatment\_duration,t.Treatment\_type

From treatment as t

Group by t.Treatment\_type;



Database management interface showing a query result for the 'treatment' table.

Buttons: Browse, Structure, SQL, Search, Insert, Export, Import

Show query box

Showing rows 0 - 2 (3 total, Query took 0.0002 seconds.)

```
Select t.Treatment_name,t.Treatment_duration,t.Treatment_type From treatment as t Gr
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

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

Extra options

				Treatment_name	Treatment_duration	Treatment_type
<input type="checkbox"/>	Edit	Copy	Delete	Angioplasty	3 Hours	Heart Disease
<input type="checkbox"/>	Edit	Copy	Delete	Dialysis	5 Hours	Kidney Disease
<input type="checkbox"/>	Edit	Copy	Delete	Chemotherapy	8 Hours	Lungs Disease

Dice :

Select d.doctor\_name,d.1st\_week,d.2nd\_week,d.3rd\_week,d.4th\_week

From doctor as d where d.Doctor\_id = 1;





The screenshot shows a database management interface with a toolbar at the top containing buttons for Browse, Structure, SQL, Search, Insert, Export, and Import. Below the toolbar is a 'Show query box' button. A green status bar indicates 'Showing rows 0 - 0 (1 total, Query took 0.0002 seconds.)'. The SQL query entered is: `Select d.doctor_name,d.1st_week,d.2nd_week,d.3rd_week,d.4th_week from doctor as d where`. Below the query is a 'Profiling' section with links for [ Edit inline ], [ Edit ], [ Explain SQL ], [ Create PHP code ], and [ Refresh ]. A control bar shows 'Show all', 'Number of rows: 25', and 'Filter rows: Search this table'. An 'Extra options' button is also present. The table data is displayed with columns: doctor\_name, 1st\_week, 2nd\_week, 3rd\_week, and 4th\_week. The first row shows 'Sairaj' with values 100, 200, 100, and 100. Below the table is another control bar with 'Show all', 'Number of rows: 25', and 'Filter rows: Search this table'.

	doctor_name	1st_week	2nd_week	3rd_week	4th_week
<input type="checkbox"/> Edit Copy Delete	Sairaj	100	200	100	100

pivot:

SELECT

doctor\_name,

MAX(CASE WHEN 1st\_week = 300 THEN 1st\_week END) AS week1, MAX(CASE

WHEN 2nd\_week = 200 THEN 2nd\_week END) AS week2, MAX(CASE WHEN

3rd\_week = 100 THEN 3rd\_week END) AS week3,

MAX(CASE WHEN 4th\_week = 100 THEN 4th\_week END) AS week4

FROM

doctor

WHERE

Doctor\_id = 1



### GROUP BY

doctor\_name;

Showing rows 0 - 0 (1 total). Query took 0.0001 seconds.

```
SELECT doctor_name, MAX(CASE WHEN 1st_week = 300 THEN 1st_week END) AS week1, MAX(CASE WHEN 2nd_week = 200 THEN 2nd_week END) AS week2,
MAX(CASE WHEN 3rd_week = 100 THEN 3rd_week END) AS week3, MAX(CASE WHEN 4th_week = 100 THEN 4th_week END) AS week4 FROM doctor WHERE
Doctor_id = 1 GROUP BY doctor_name;
```

☐ Profiling [\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Explain SQL \]](#) [\[ Create PHP code \]](#) [\[ Refresh \]](#)

☐ Show all | Number of rows: 25 | Filter rows:

[Edit options](#)

	doctor_name	week1	week2	week3	week4
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a> <a href="#">Save</a>		NULL	200	100	100

### Conclusion:

The implementation of dimension and fact tables, along with OLAP operations, is crucial for building a robust and efficient data analytics environment. It empowers organizations to extract valuable insights from their data, make data-driven decisions, and adapt to changing business needs. However, it's important to continuously monitor and maintain the data warehouse to ensure data accuracy and relevance.