**INDEX**

**Topics**: **Page No**

1. Index 1
2. Problem Statement 2
3. ERD 3
4. List of Operational Tables Created 4
5. Contents of all Operational Tables 5
6. Information Package Diagram 9
7. Start Schema 10
8. Procedures for Data Loading 11
9. Contents of DW tables 13
10. OLAP Queries and output 16

**Problem Statement**

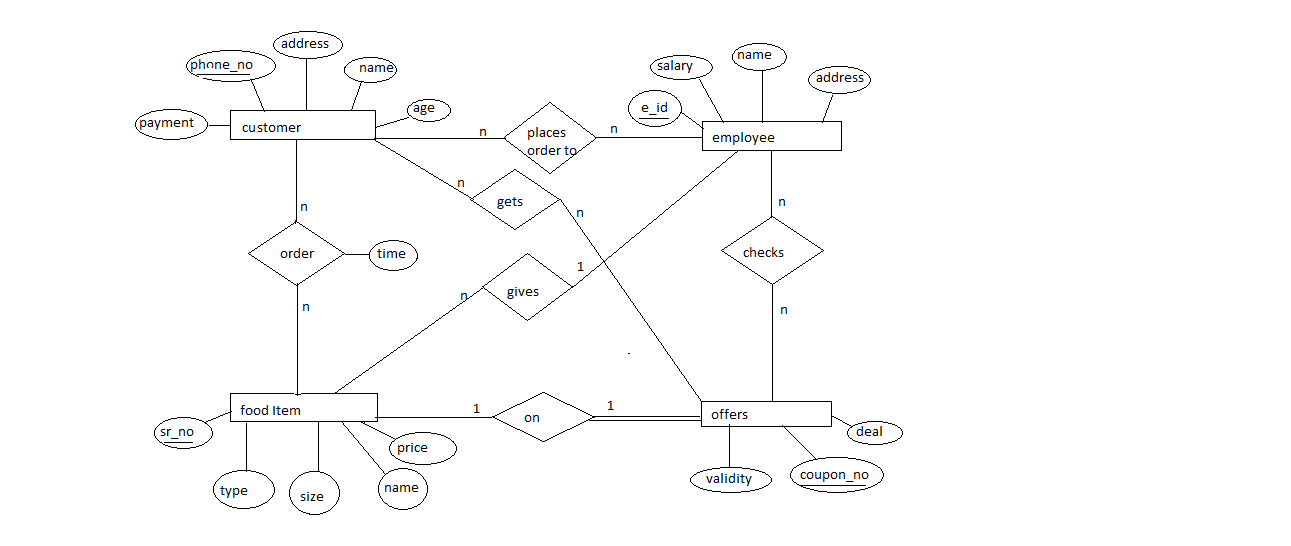
Dominoes Pizzas is India's fastest growing fast food service restaurant for casual and fine dining. The current Dominoes menu features a variety of Italian and American entrees. Pizza being the primary focus is available in traditional, speciality and custom pizzas in variety of crust styles and toppings. Additional entrees include pasta, bread-bowls, desserts etc.

**Dominoes Pizzas** – “KhushiyokiHome Delivery”.

**Analysis**

Dominoes Pizzas will consider sales in consideration to other dimensions to carry out required analysis for the proposed expansion. Sales with respect to time will be considered. Also in consideration will be sales with respect to location.Example: Certain food delicacy will have higher sales in certain region and other delicacy for other region.Sales with respect to customers are also another important factor under our analysis. Example: Age groups and Gender ratio of customers will be taken into consideration. Another important parameter to be considered is sales with respect to specific products.

**ERD**

****

**List of Operational Tables Created**

* Customer
* Employee
* food\_item
* offer
* place\_order
* order1
* gets
* checks

**Contents of all Operational Tables**

* SQL> select \* from customer;

PHONE\_NO NAME ADDRESS AGE PAYMENT

---------- ---------- ---------- ---------- ----------

9892524750 Aum borivali 20 700

9892522650 Glancy vasai 10 500

9892524640 Albina vasai 40 800

9892524650 rohit borivali 70 1000

* SQL> select \* from employee;

E\_ID NAME ADDRESS SALARY

---------- ---------- ---------- ----------

1 Jinson kandivali 10000

2 harshal dahisar 9000

3 frenia andheri 8000

4 carl malad 12000

* SQL> select \* from food\_item;

SR\_NO NAME TYPE SIZE1 PRICE E\_ID

---------- ---------- ---------- ---------- ---------- ----------

10 pizza veg medium 140 1

11 pizza nveg large 240 2

13 pizza veg medium 140 1

14 garlic veg medium 80 2

15 pizza veg medium 140 1

16 coke veg medium 40 3

17 pasta nveg large 140 3

18 pizza veg medium 140 4

19 pasta veg small 80 3

20 coke veg medium 40 4

* SQL> select \* from offer;

COUPNE\_NO VALIDITY DEAL SR\_NO

---------- --------- ---------- ----------

100 23-FEB-13 10% off\* 10

200 30-JAN-13 15% off\* 14

300 10-FEB-13 cokefree\* 15

400 03-MAR-13 buy1get1\* 16

* SQL> select \* from place\_order;

PHONE\_NO E\_ID

---------- ----------

9892522650 2

9892524640 1

9892524650 3

9892524650 4

9892524750 2

9892524750 4

* SQL> select \* from order1;

PHONE\_NO SR\_NO

---------- ----------

TIME1

---------------------------------------------------------------------------

9892524650 15

08-JAN-13 10.42.24.000000 AM

9892524750 10

03-JAN-13 11.23.34.000000 AM

9892522650 18

26-JAN-13 04.26.14.000000 AM

PHONE\_NO SR\_NO

---------- ----------

TIME1

---------------------------------------------------------------------------

9892524650 17

08-JAN-13 10.42.24.000000 AM

9892524640 11

15-JAN-13 09.28.54.000000 AM

* SQL> select \* from gets;

PHONE\_NO COUPNE\_NO

---------- ----------

9892522650 400

9892524640 200

9892524650 300

9892524750 100

* SQL> select \* from checks;

E\_ID COUPNE\_NO

---------- ----------

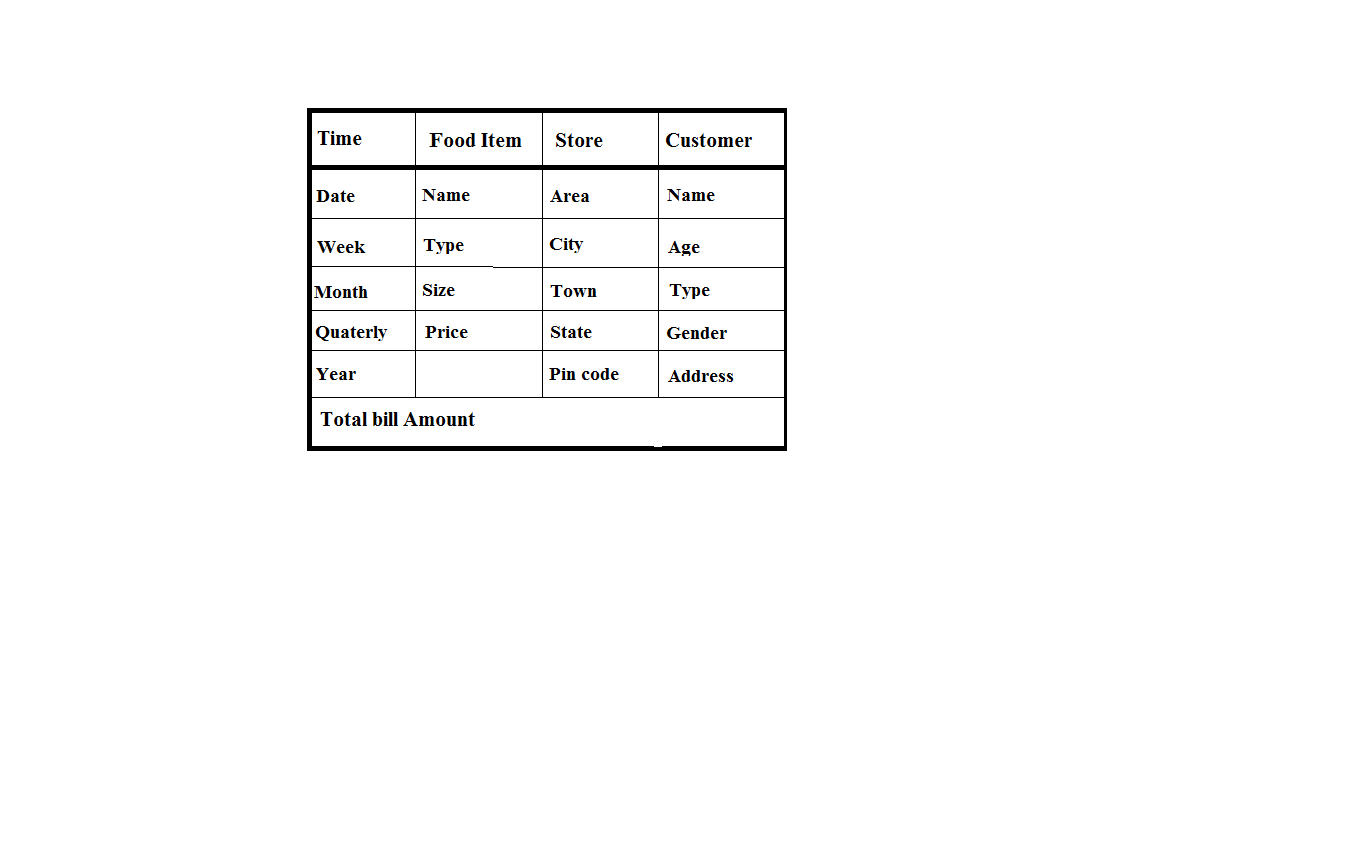
1 200

2 400

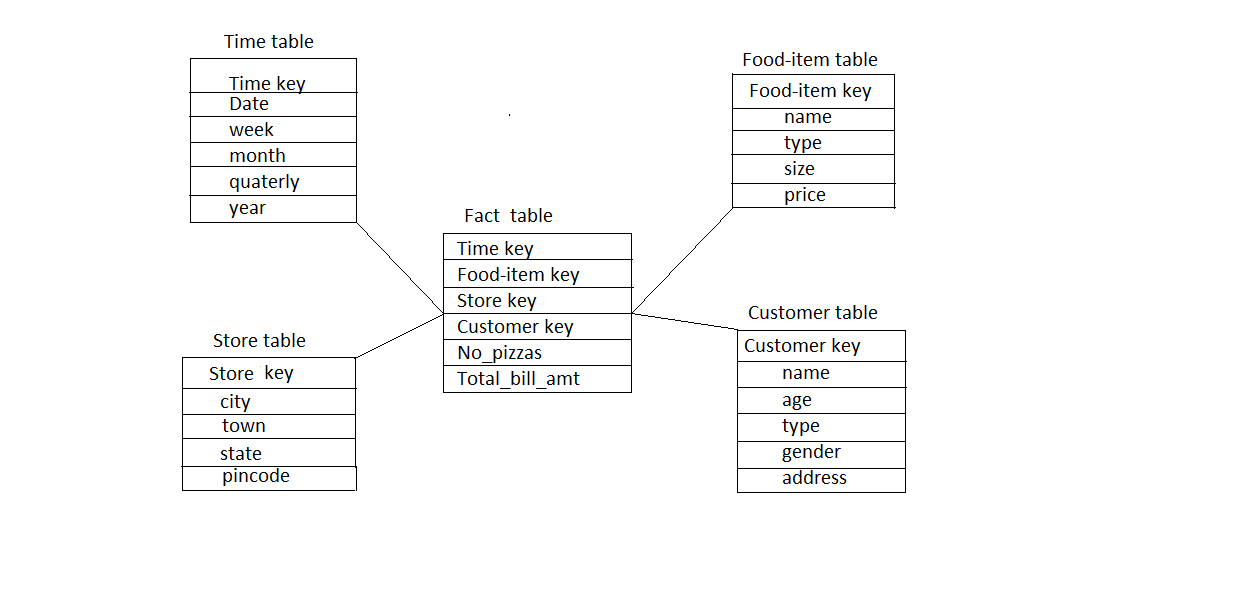
3 300

4 100

**Information Package Diagram**

****

**Start Schema**

****

**Procedures for Data Loading**

SQL> Declare

2 variable number;

3 id number;

4 name1 varchar2(100);

5 type1 varchar2(100);

6 size2 varchar2(100);

7 price1 number;

8

9 CURSOR C1 IS

10 select food\_item.sr\_no variable

11 from food\_item;

12

13 BEGIN

14 FOR R\_C1 IN C1 LOOP

15 BEGIN

16 select food\_item.sr\_no,food\_item.name,food\_item.type,food\_item.size1,food\_i

tem.price

17 into id,name1,type1,size2,price1

18 from food\_item

19 where food\_item.sr\_no=R\_C1.variable;

20 ----If exists then update otherwise insert

21 INSERT INTO fooditem1(food\_itemkey,name,type,size1,price) values(id,name1,t

ype1,size2,price1);

22 dbms\_output.put\_line(id ||''||'Already Exists');

23 EXCEPTION

24 WHEN DUP\_VAL\_ON\_INDEX THEN

25 dbms\_output.put\_line(R\_C1.variable);

26 END;

27 END LOOP;

28 END;

29

30 /

PL/SQL procedure successfully completed.

**Contents of DW tables**

* SQL> select \* from customer1;

CUSTOMERKEY NAME ADDRESS AGE PAYMENT

----------- ---------- ---------- ---------- ----------

9637142573 flevia jaipur 30 600

8983436824 stalen hyderabad 40 900

9096690209 flavian bangalore 20 800

9892524750 Aum borivali 20 700

9892522650 Glancy vasai 10 500

9892524640 Albina vasai 40 800

9892524650 rohit borivali 70 1000

* SQL> select \* from fooditem1;

FOOD\_ITEMKEY NAME TYPE SIZE1 PRICE

------------ ---------- ---------- ---------- ----------

10 pizza veg medium 140

11 pizza nveg large 240

13 pizza veg medium 140

15 pizza veg medium 140

16 coke veg medium 40

17 pasta nveg large 140

18 pizza veg medium 140

19 pasta veg small 80

20 coke veg medium 40

* SQL> select \* from time1;

TIMEKEY DATE1 WEEK MONTH QUATER YEAR

---------- --------- ---------- ---------- ---------- ----------

51 08-JAN-13 1 1 1 2013

52 03-JAN-13 1 1 1 2013

53 26-JAN-13 4 1 1 2013

54 08-JAN-13 1 1 1 2013

55 15-JAN-13 3 1 1 2013

56 17-FEB-12 3 2 1 2012

57 05-MAR-13 1 3 1 2013

58 25-APR-13 3 4 1 2013

* SQL> select \* from store1;

STOREKEY CITY TOWN STATE PINCODE

---------- ---------- ---------- ---------- ----------

35 Bangalore town2 Karnataka 512002

76 Hyderabad town3 Andhra 415802

98 Jaipur town4 Rajasthan 441502

19 Kolkata town5 WestBengal 184522

50 Borivali town1 Maharash 400002

49 vasai town6 Maharash 400002

* SQL> select \* from fact1;

TIMEKEY FOOD\_ITEMKEY STOREKEY CUSTOMERKEY TOTAL\_AMT

---------- ------------ ---------- ----------- ----------

51 15 50 9892524650 1000

52 10 50 9892524750 700

53 18 49 9892522650 500

54 17 50 9892524650 1000

55 11 49 9892524640 800

56 13 98 9637142573 600

57 15 76 8983436824 900

58 19 35 9096690209 800

56 19 50 9892524650 200

53 11 49 9892524750 400

**OLAP Queries and output**

* **Total Sales Recorded**

SQL> select sum(total\_amt) as total\_amount from fact1;

TOTAL\_AMOUNT

------------

6300

* **Purchase by each customer**

SQL> SELECT customerkey,

2 COUNT(\*) AS num\_rows,

3 SUM(total\_amt) AS sales\_value

4 FROM fact1

5 GROUP BY customerkey;

CUSTOMERKEY NUM\_ROWS SALES\_VALUE

----------- ---------- -----------

9892524750 2 1100

9892522650 1 500

9892524640 1 800

8983436824 1 900

9892524650 3 2200

9637142573 1 600

9096690209 1 800

7 rows selected.

* **Total sales per location and time period**

SQL> SELECT storekey,

2 timekey,

3 COUNT(\*) AS num\_rows,

4 SUM(total\_amt) AS total\_value

5 FROM fact1

6 GROUP BY storekey, timekey

7 ORDER BY storekey, timekey;

STOREKEY TIMEKEY NUM\_ROWS TOTAL\_VALUE

---------- ---------- ---------- -----------

35 58 1 800

49 53 2 900

49 55 1 800

50 51 1 1000

50 52 1 700

50 54 1 1000

50 56 1 200

76 57 1 900

98 56 1 600

9 rows selected.

* **Total sales per location and time with SubTotals (Rollups)**

SQL> SELECT storekey,

2 timekey,

3 SUM(total\_amt) AS total\_value

4 FROM fact1

5 GROUP BY ROLLUP (storekey, timekey)

6 ORDER BY storekey, timekey;

STOREKEY TIMEKEY TOTAL\_VALUE

---------- ---------- -----------

35 58 800

35 800

49 53 900

49 55 800

49 1700

50 51 1000

50 52 700

50 54 1000

50 56 200

50 2900

76 57 900

STOREKEY TIMEKEY TOTAL\_VALUE

---------- ---------- -----------

76 900

98 56 600

98 600

6900

15 rows selected.

* **Total sales per location and time with SubTotals (Cube)**

SQL> SELECT storekey,

2 timekey,

3 SUM(total\_amt) AS total\_value

4 FROM fact1

5 GROUP BY CUBE (storekey, timekey)

6 ORDER BY storekey, timekey;

STOREKEY TIMEKEY TOTAL\_VALUE

---------- ---------- -----------

35 58 800

35 800

49 53 900

49 55 800

49 1700

50 51 1000

50 52 700

50 54 1000

50 56 200

50 2900

76 57 900

STOREKEY TIMEKEY TOTAL\_VALUE

---------- ---------- -----------

76 900

98 56 600

98 600

51 1000

52 700

53 900

54 1000

55 800

56 800

57 900

58 800

STOREKEY TIMEKEY TOTAL\_VALUE

---------- ---------- -----------

6900

23 rows selected.

* **Total sale amount at location ‘vasai’**

SQL> SELECT SUM(s.total\_amt) AS total\_sale

2 from fact1 s, store1 l

3 wheres.storekey=l.storekey and l.city='vasai';

TOTAL\_SALE

----------

1700

* **Total sale amount in 2013**

SQL> SELECT SUM(total\_amt) AS total\_sale

2 from fact1 s, time1 t

3 wheres.timekey=t.timekey and t.year=2013;

TOTAL\_SALE

----------

6100

* **No of Products of the name pasta**

SQL> SELECT COUNT(\*) AS no\_products

2 from fact1 s, fooditem1 p

3 wheres.food\_itemkey=p.food\_itemkey and p.name='pasta';

NO\_PRODUCTS

-----------

3

* **Monthly sales for the year 2013**

SQL> SELECT t.month, SUM(total\_amt)AS monthly\_sales

2 from fact1 s, time1 t

3 wheret.year=2013 and s.timekey=t.timekey

4 GROUP BY t.month

5 ORDER BY t.month;

MONTH MONTHLY\_SALES

---------- -------------

1 4400

3 900

4 800

* **Quarterly Sales Info for the year 2012 and 2013.**

SQL> SELECTt.year, t.quater, SUM(s.total\_amt)AS quarterly\_sales, count(\*) AS N

o\_of\_sales

2 from fact1 s, time1 t

3 wheret.year IN(2012,2013) and s.timekey=t.timekey

4 GROUP BY t.quater, t.year

5 ORDER BY t.year, t.quater;

YEAR QUATER QUARTERLY\_SALES NO\_OF\_SALES

---------- ---------- --------------- -----------

2012 1 800 2

2013 1 6100 8