Amrit Pandey

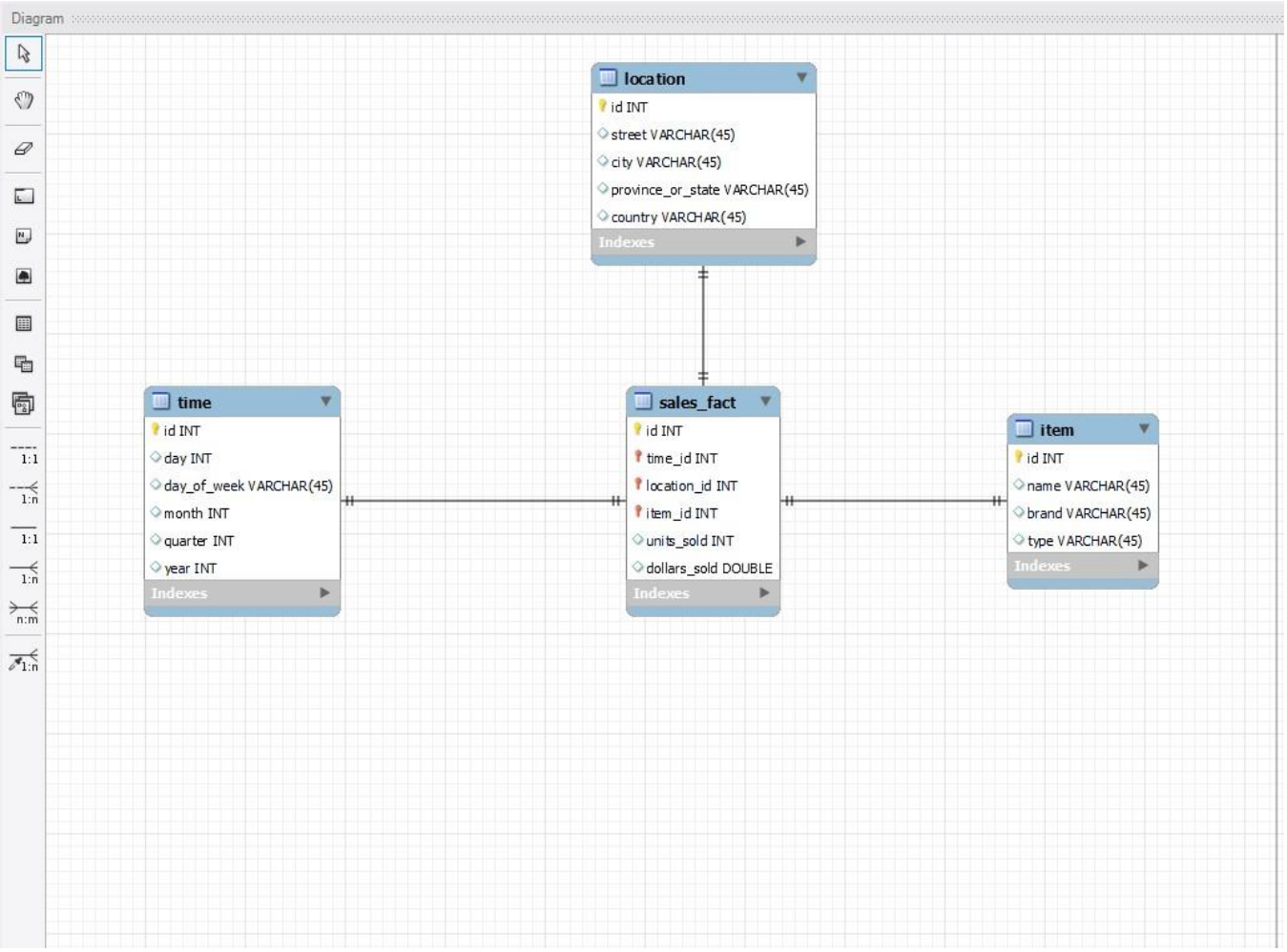
207907

MCA, SEM II, YEAR II

Knowledge Engineering Lab

Solution 1

Star Schema for the Data Warehouse:



a. **“Compute the sum of sales, grouping by city and item.**

a.sql

SQL Worksheet History

Find

Worksheet Query Builder

```
select sum(s.units_sold), l.city, i.name as Item_Name
from sales_fact s, location l, item i
where s.location_key = l.id and s.item_key = i.id
group by l.city, i.name;
```

Script Output Query Result

SQL | All Rows Fetched: 8 in 0.003 seconds

	SUM(S.UNITS_SOLD)	CITY	ITEM_NAME
1	2000	Palo Alto	Iphone X13
2	2600	Palo Alto	Iphone S
3	10600	Palo Alto	Galaxy Tab A7
4	2000	Palo Alto	Galaxy Note 13
5	2000	Palo Alto	MacBook
6	2000	Palo Alto	HP 15x
7	4000	NYC	Iphone X13
8	1000	NYC	Galaxy Note 13

a.sql

SQL Worksheet History

Find

Worksheet Query Builder

```
select sum(s.units_sold), l.city, i.name as Item_Name
from sales_fact s, location l, item i
where s.location_key = l.id and s.item_key = i.id
group by cube (l.city, i.name);
```

Script Output Query Result

SQL | All Rows Fetched: 17 in 0.004 seconds

	SUM(S.UNITS_SOLD)	CITY	ITEM_NAME
1	26200	(null)	(null)
2	2000	(null)	HP 15x
3	2000	(null)	MacBook
4	2600	(null)	Iphone S
5	6000	(null)	Iphone X13
6	10600	(null)	Galaxy Tab A7
7	3000	(null)	Galaxy Note 13
8	5000	NYC	(null)
9	4000	NYC	Iphone X13
10	1000	NYC	Galaxy Note 13
11	21200	Palo Alto	(null)
12	2000	Palo Alto	HP 15x
13	2000	Palo Alto	MacBook
14	2600	Palo Alto	Iphone S
15	2000	Palo Alto	Iphone X13
16	10600	Palo Alto	Galaxy Tab A7
17	2000	Palo Alto	Galaxy Note 13

a. **“Compute the sum of sales, grouping by city.**

a.sql

SQL Worksheet | History

Find

Worksheet | Query Builder

```
select sum(s.units_sold), l.city
from sales_fact s, location l, item i
where s.location_key = l.id and s.item_key = i.id
group by city;
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.006 seconds

	SUM(S.UNITS_SOLD)	CITY
1	21200	Palo Alto
2	5000	NYC

a.sql

SQL Worksheet | History

Find

Worksheet | Query Builder

```
select sum(s.units_sold), l.city
from sales_fact s, location l, item i
where s.location_key = l.id and s.item_key = i.id
group by cube(city);
```

Script Output | Query Result

SQL | All Rows Fetched: 3 in 0.002 seconds

	SUM(S.UNITS_SOLD)	CITY
1	26200	(null)
2	5000	NYC
3	21200	Palo Alto

b. "Compute the sum of sales, grouping by item.

a.sql

SQL Worksheet | History

Find

Worksheet | Query Builder

```
select sum(s.units_sold), i.name as Item_Name
from sales_fact s, item i
where s.item_key = i.id
group by i.name;
```

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.005 seconds

	SUM(S.UNITS_SOLD)	ITEM_NAME
1	6000	Iphone X13
2	2600	Iphone S
3	10600	Galaxy Tab A7
4	3000	Galaxy Note 13
5	2000	MacBook
6	2000	HP 15x

a.sql

SQL Worksheet | History

Find

Worksheet | Query Builder

```
select i.name, sum(units_sold) from
sales_fact s, item i
where s.item_key = i.id
group by cube(i.name);
```

Script Output | Query Result

SQL | All Rows Fetched: 7 in 0.004 seconds

	NAME	SUM(UNITS_SOLD)
1	(null)	26200
2	HP 15x	2000
3	MacBook	2000
4	Iphone S	2600
5	Iphone X13	6000
6	Galaxy Tab A7	10600
7	Galaxy Note 13	3000

d. What is the maximum number of cells in base cuboid?

Ans: Maximum number of cells in base cuboid =

Values of Date * Values of Game * Values of Spectator * Values of Location

= (30 * 12 * 10) * (1) * (3) * (2) [assuming on 10 years in date]

= 21,600

e. What is the minimum number of cells in base cuboid?

Ans: Minimum number of cells in base cuboid =

= Maximum(Values of Date, Values of Game, Values of Spectator, Values of Location)

= Maximum((30 * 12 * 10), 1, 3, 2))

= 3600

i.e. atleast 3600 tuples are needed to store all distinct values of date.

Solution 2

a. And implement that operation using OLAP query language.

a.sql

SQL Worksheet | History

Worksheet | Query Builder

```sql
select d.year, s.type, l.city, sum(charge) from fact f, "date" d, spectator s, location l
where
    f.spectator_spectator_id = s.spectator_id and
    f.date_date_id = d.date_id and
    f.location_location_id = l.location_id and
    s.type = 'student' and
    l.city = 'GM_Place' and
    d.year = 2010
group by
    rollup (d.year, s.type, l.city);
```

Script Output | Query Result

SQL | All Rows Fetched: 4 in 0.005 seconds

	YEAR	TYPE	CITY	SUM(CHARGE)
1	2010	student	GM_Place	110
2	2010	student	(null)	110
3	2010	(null)	(null)	110
4	(null)	(null)	(null)	110

b. Perform rollup operation from date to year

a.sql

SQL Worksheet History

Worksheet Query Builder

```
select count, charge, d.day from fact f, "date" d
where f.date_date_id = d.date_id;
```

Script Output Query Result

SQL | All Rows Fetched: 7 in 0.005 seconds

	COUNT	CHARGE	DAY
1	10	22	22
2	10	22	22
3	10	22	22
4	10	22	22
5	10	22	22
6	10	22	23
7	10	22	23

After rollup:

a.sql

SQL Worksheet History

Worksheet Query Builder

```
select sum(count) as count, sum(charge) as charge, d.year from fact f, "date" d
where f.date_date_id = d.date_id
group by rollup(d.year);
```

Script Output Query Result

SQL | All Rows Fetched: 3 in 0.004 seconds

	COUNT	CHARGE	YEAR
1	50	110	2010
2	20	44	2011
3	70	154	(null)

c. **What is the average charge paid by students, adults and seniors in each category you need to compute average?**

a.sql

SQL Worksheet History

Worksheet Query Builder

```sql
select spectator_type, category_type, avg(charge) as AvgCharge
    from
        fact f,
        (select
            s.type as spectator_type,
            s.spectator_id as spec_id,
            c.type as category_type
            from
                spectator s
                    inner join category c
                        on s.category_cat_id = c.cat_id)
where
    f.spectator_spectator_id = spec_id
group by
    (spectator_type, category_type);
```

Script Output Query Result

SQL | All Rows Fetched: 3 in 0.011 seconds

	SPECTATOR_TYPE	CATEGORY_TYPE	AVGCHARGE
1	student	eco	34
2	student	gold	34
3	adult	gold	34

d. Draw the snowflake schema diagram for the data warehouse.