ER and Relational Modeling

1. Given an ER diagram there is only one possible equivalent Relational model.
2. Weak entities don't have a primary key.
3. ISA hierarchies can not be represented by a single table
4. A table representing a relationship in relational model has a multi-column candidate key
5. A candidate key can not be a primary key
6. Reference Integrity is enforced by Primary Key - Foreign Key relationship definition
7. Given a set of functional dependencies we can always create a relational schema in BCNF.
8. A Star schema dimension table is usually in 3NF
9. The normalization process using functional dependencies is essentially same as finding entities and relationship in the ER model

RA

- Write the relational algebra expression equivalent to:

```
SELECT ProdBrand
FROM Product, Sales
WHERE Product.ProdId = Sales.ProdId and Sales.Amount > 100
INTERSECT
SELECT ProdBrand
FROM Product, Sales
WHERE Product.ProdId = Sales.ProdId and Price > 10
```

- $\pi_{\text{ProdBrand}}(\sigma_{\text{amount} > 100}(\text{Sales}) \bowtie_{\text{prodid}} (\text{Product}) \cap \pi_{\text{ProdBrand}}(\sigma_{\text{price} > 10}(\text{Product}) \bowtie_{\text{prodid}} \text{Sales})$

SQL

1. Show the output of the following query given the two tables T1 and T2:

```
SELECT orig, sum(newb),
   sum(sum(newb)) over (order by orig rows
unbounded preceding),
   count(DISTINCT a)
FROM (SELECT 'From t1', a, b
      FROM t1
      UNION ALL
      SELECT 'From t2', a, b
      FROM t2
      UNION ALL
      SELECT 'From t3', 5, 50) d1(orig,a,newb)
GROUP BY orig;
```

T1

A	**B**
1	10
1	20
2	30
3	40

T2

A	**B**
1	20
2	10
3	40
4	40

orig	**A**	**newB**
FromT1	1	10
FromT1	1	20
FromT1	2	30
FromT1	3	40
FromT2	1	20
FromT2	2	10
FromT2	3	40
FromT2	4	40
FromT3	5	50

orig		**sum(newB)**
FromT1	1	100
FromT2	4	110
FromT3	5	50

orig		**sum(newB)**	**cumulative**
FromT1	1	100	100
FromT2	4	110	210
FromT3	5	50	260

Storage

1. If there is no room left on the track of a HDD, what's the next best place to put the next disk block

2. Disk Mirroring is another name for which RAID level

3. Main purpose of data striping is to improve what aspect of the I/O system

4. A 16 Bit block is striped across 4 disks in 4 bit blocks as [0100][1110][1010][0010]. What's the value of the parity block

5. the best replacement policy for a database buffer pool manager is

Index

(RID)	**Supplier**	**Part**	**Price**
1001	Ace	Nails	0.05
1002	Acme	Bolts	0.10
1003	Ace	Nuts	0.05
1004	Amzn	Screws	0.20
2001	Ace	Nuts	0.08
2002	OurParts	Nuts	0.04
2004		Nuts	
3001	CheapDeals		
3002	OurParts	Nails	
3003	Ace	Bolts	0.08

1. Write the index on column Part in the tabular form.

2. Is it a unique or a non-unique index?

3. Write the SQL to create this index.

Extendible hashing

- 0: 2,20,20,24,32 => needs split
- 1: 7,11,13,15,17,19,21,23,27,35 => needs split
- So: global depth = 2, local depth = 2 in each case
- 00: 20,20, 24, 32
- 01: 13, 17, 21
- 10: 2
- 11: 7,11,15,19,23,27,35 => needs split
- => 11 would need a split, global depth = 3, and local depth for these = 3
- 011: 11,19,27,35
- 111: 7,15,23

Query 1

Sales(ProdId, StoreId, SalesDate, Amount), ProdId and StoreId are FK's
Product(ProdId, ProdName, ProdBrand, Price), PK: ProdiId
Store(StoreId, StoreName, StoreCity, State), PK: StoreId

1.Express in SQL: For each store, show the product id and name that had the
  maximum sales.

```
/* first find the sales by product and store */
with step1(storeId, prodId, total_sales) as
(select storeId, prodId, sum(amount)
from sales
group by storeId, prodId),
/* then find the maxsales (across products) in each store */
step2(storeId, maxsales) as
(select storeId, max(total_sales)
 from step1
 group by storeId)
/* now find the product with maxsales for each store */
```

Alternatives

```
select s.storeid, s.storename, p.prodid, p.prodname,step1.total_sales
from step1, product p, store s
where step1.storeid = s.storeid and step1.prodid = p.prodid and
      (step1.storeid, step1.total_sales) in
      (select storeid, maxsales from step2);

select s.storeid, s.storename, p.prodid, p.prodname, step1.total_sales
from step1, product p, store s
where step1.storeid = s.storeid and step1.prodid = p.prodid and
      step1.total_sales in
          (select maxsales
           from step2
           where step1.storeid = step2.storeid);

select s.storeid, s.storename, p.prodid, p.prodname, step1.total_sales
from step1, product p, store s, step2
where step1.storeid = s.storeid and step1.prodid = p.prodid and
      step1.total_sales = step2.maxsales
      and step1.storeid = step2.storeid;
```

Query 2

Express in SQL: Show the top 3 stores by sales in each state.

```
/* determine total_sales for each store */
with step1(state, storename, storeId, total_sales) as
(select state, storename, sales.storeid, sum(amount)
from sales,store
where sales.storeid = store.storeid
group by state, storename, sales.storeId),
/* determine rank in each state  */
step2(state, storename, storeid, total_sales, sales_rank) as
(select state, storename, storeid, total_sales,
      rank() over (partition by state order by total_sales desc)
 from step1)
/* choose top 3 */
select *
from step2
where sales_rank <= 3;
```