

Chapter 8

- Slowly Changing Dimensions (SCDs)
 - When data changes in the operational systems the change can be propagated to the warehouse in many ways
 - Data warehousing has developed some ‘standard’ approaches:
 - Type 1, type 2, type 3, hybrids, time-stamped dimensions, and tracking changes via fact tables

Chapter 8

- Consider the Northwind OLTP
 - Product unit price is recorded in the Products table. This is the current price.
 - What happens to the history of price changes
 - Partial information is in the OrderDetails table: we can construct the pricing history of a product, but that can be incomplete ...
- What could we do in our Star Schema to keep a full account of product data?

Tracking Changes via Fact Table

Fig 8-2 shows how a fact table can be used to capture complete information on changes to a dimension

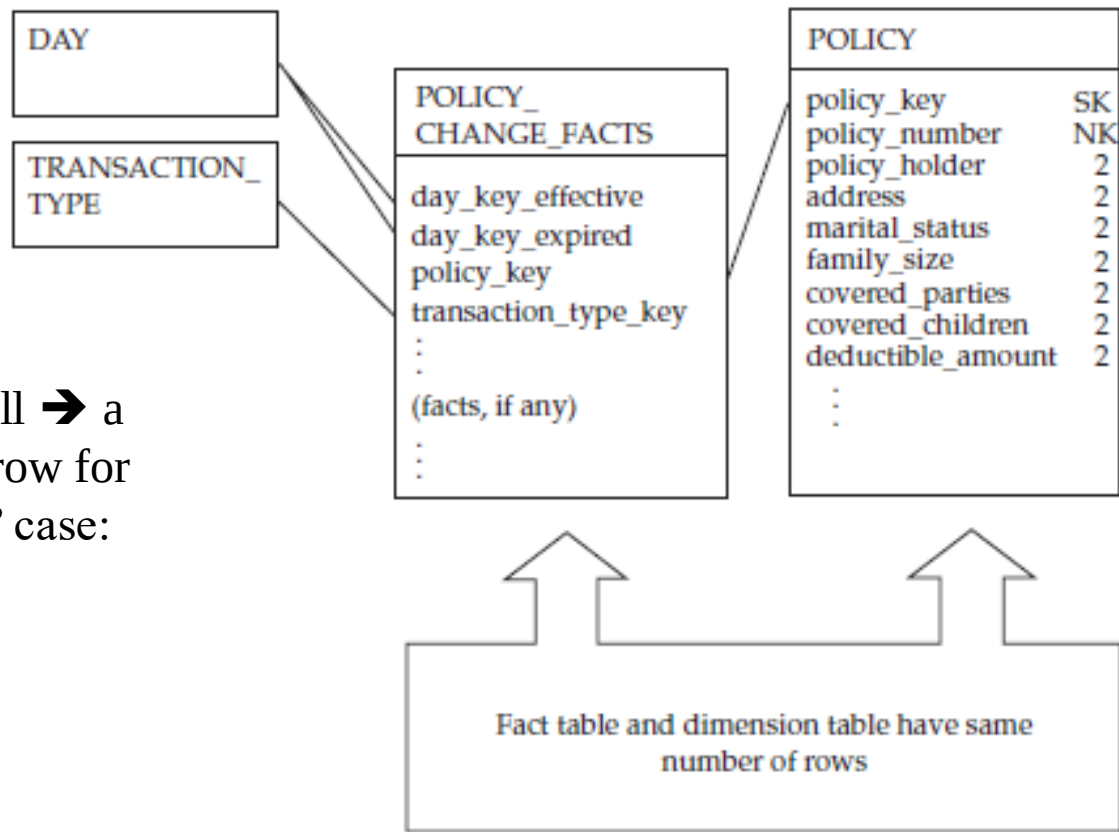
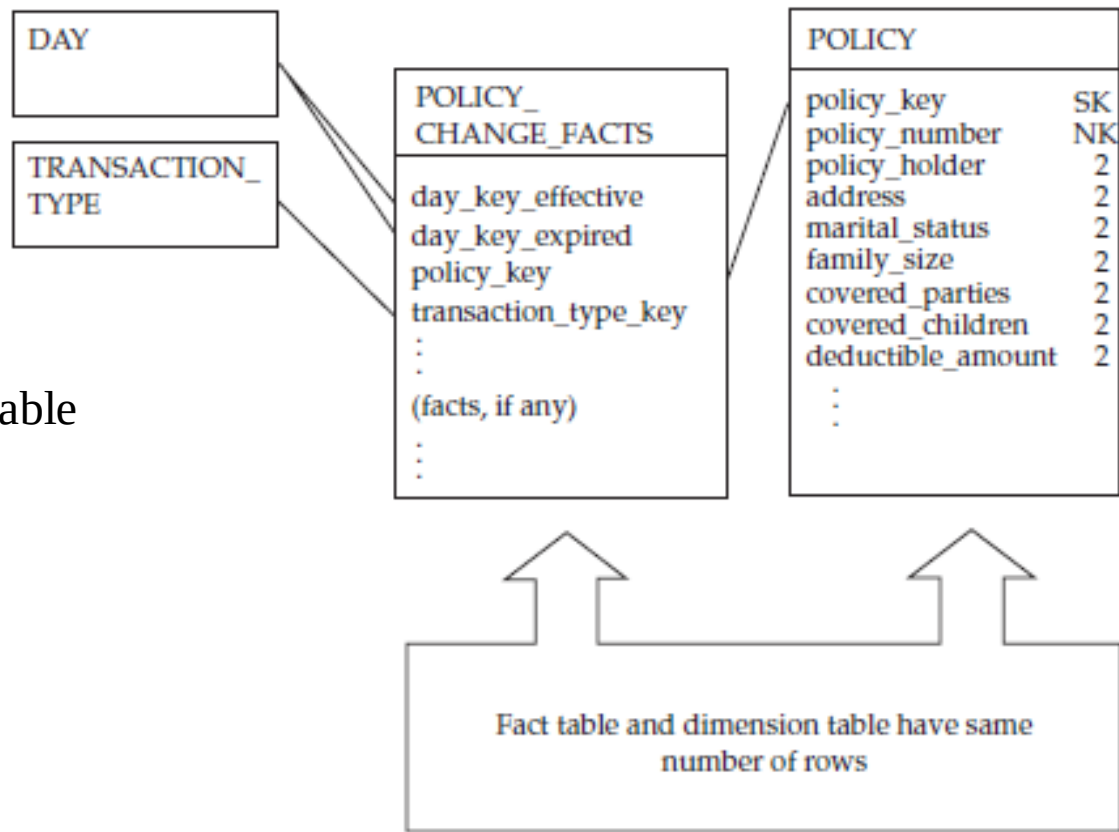


Figure 8-2 A fact table records the change history of the policy dimension

Tracking Changes via Fact Table

Fig 8-2 shows how a fact table can be used to capture complete information on changes to a dimension



One row in fact table
for each row in
dimension !!!

Figure 8-2 A fact table records the change history of the policy dimension

Tracking History via Type 1

If something changes in the operational system then a Type 1 response is a simple overwrite in the warehouse

No history kept ... only most current value

May be appropriate for some attributes

Tracking History via Type 2

Most common approach, but note the structure of type 2 examples so far in the text ... fig 8-1

The design does not let us get a full picture of anyone's policy at a particular point in time.

Through the connection to Payment_Facts we can get partial information.

Type 2

Time-stamped Type 2

POLICY
policy_key
policy_number
policy_holder
marital_status
family_size
covered_parties
spouse_coverage
covered_children
deductible_amount
:
:
transaction_type
effective_date
expiration_date
most_recent_version

← Warehouse key

← Reason for change

Effective & expiration dates (no nulls)

← A current flag/indicator

Type 2

Time-stamped Type 2

POLICY

policy_key	policy_number	policy_holder	transaction_type	effective_date	expiration_date	most_recent_version	marital_status	family_size	covered_parties	
12882	40111	Smith, Hal	New Policy	2/14/2005	2/11/2006	Expired	Single	1	1	
12911	40111	Smith, Hal	Policy Change	2/12/2006	3/30/2006	Expired	Married	2	1	
13400	40111	Smith, Hal	Policy Renewal	3/31/2006	12/19/2007	Expired	Married	2	2	
14779	40111	Smith, Hal	Policy Change	12/20/2007	2/3/2008	Expired	Married	3	3	
14922	40111	Smith, Hal	Policy Change	2/4/2008	12/31/9999	Current	Married	4	4	

Use to order a change history

Use for point-in-time analysis across policies

Use to filter for current status

Type 2

Time-stamped Type 2

```
SELECT
  policy_holder,
  transaction_type,
  marital_status
  :
ORDER_BY
  effective_date
```

*A picture of a
policy holder
over time*

```
SELECT
  policy_holder,
  marital_status
  :
WHERE
  12/31/2006 >= effective_date AND
  12/31/2006 <= expiration_date
```

*A picture of a policy
holder at a point in time*

```
SELECT
  policy_holder,
  marital_status
  :
WHERE
  most_recent_row
    = "Current"
```

*A picture of a
policy holder at
the current time*

Type 2

Time-stamped Type 2

Suppose there are multiple changes on one day

A variation:

- ... utilize another dimension for time ...
- ... utilize another flag for last_change_of_day
and having possible values *final* / *expired*

Type 2

Some refer to “time-stamped type 2” is “**the** type 2” :
A dimension with *effective & expired* dates plus a *current* indicator.

In 4904 we only concern ourselves with Type 2 having *effective & expired* dates plus a *current* indicator

Tracking History via Type 3

Occasionally an attribute of a dimension has special analytic requirements

Analysts want to understand some business process as things were before a change, and then as well after the change.

For such attributes we include 2 attributes, such as
region_current *region_previous*

Type 3

Legend:

SK Surrogate Key

NK Natural Key

1 Type 1

2 Type 2

3 Type 3 Pair

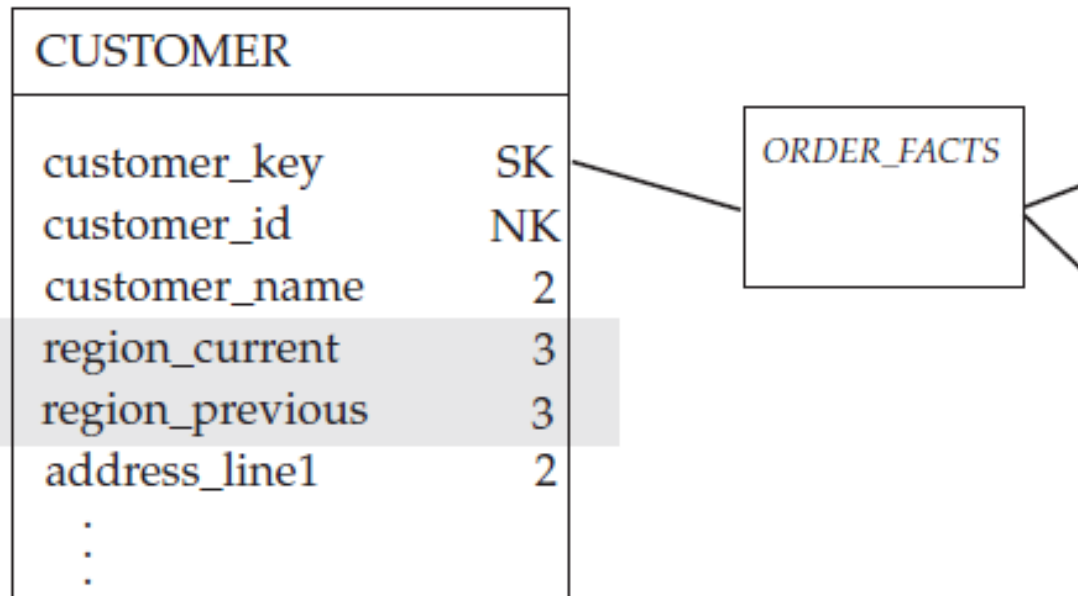


Figure 8-4 A type 3 attribute in the customer dimension

Type 3

Before

Customer dimension table

customer_ key	customer_ID	customer_name	region_ current	region_ previous
1011	1140400	Davis, Robert	East	East
1022	3305300	Nguyen, Tamara	East	East
1302	7733300	Rodriguez, Jason	West	West
1499	9900011	Johnson, Sue	West	West

initially

Orders fact table

customer_ key	day_key	order_ dollars
1011	2322	2000
1011	3422	1000

Type 3

After

Customer dimension table

customer_key	customer_ID	customer_name	region_current	region_previous
1011	1140400	Davis, Robert	Northeast	East
1022	3305300	Nguyen, Tamara	Southeast	East
1302	7733300	Rodriguez, Jason	West	West
1499	9900011	Johnson, Sue	West	West

Later on

Analysis
with
new values

Analysis
with
old values

Orders fact table

customer_key	day_key	order_dollars
1011	2322	2000
1011	3422	1000
1011	6599	1200
1011	8211	2000

} facts in place before the change

} facts added after the change

Type 3

SQL for total sales by Region

by regionCurrent

```
Select regionCurrent,  
       sum(orderDollars)  
From Customer ....join Sales  
       on(...)  
Group by regionCurrent
```

by regionCurrent

```
Select regionCurrent,  
       sum(orderDollars)  
From Customer ....join Sales  
       on(...)  
Group by regionCurrent
```

How would you do the above if the dimension was Type 2

...by regionCurrent	...by regionPrevious
...doable	...more difficult

Type 3

Variations 3 attributes instead of 2

regionCurrent, regionLastYear, regionTwoYearsAgo

Hybrid
SCD

CUSTOMER	
customer_key	SK
company_id	NK
company_name_current	1
company_name_historic	2
⋮	

ORDER_FACTS



Type 1 / 2 Hybrid

CUSTOMER

customer_ key	company_ id	company_name _current	company_name _historic
1011	BB770	Apple Computer, Inc.	Apple Computer, Inc.



Name changes to Apple Inc.



CUSTOMER

customer_ key	company_ id	company_name _current	company_name _historic
1011	BB770	Apple Computer, Inc. Apple Inc. 1	Apple Computer, Inc.
2 2822	BB770	Apple Inc.	Apple Inc.

1 Old row(s) updated with new company_name_current

2 Row is added with new name in both positions

Type 1 / 2 Hybrid

CUSTOMER

customer_ key	customer_ id	company_name _current	company_name _historic
1011	BB770	Apple Computer, Inc. Apple Inc.	Apple Computer, Inc.
2822	BB770	Apple Inc.	Apple Inc.

ORDER_FACTS

customer_ key	date_ key	order_ dollars
1011	1211	200
1011	1221	400
2822	1344	400

} Facts in place before change

} Fact added after change

Type 1 / 2 Hybrid

Current info

```
SELECT
  company_name_current,
  sum(order_dollars)
FROM
  :
```



<u>COMPANY_ NAME_CURRENT</u>	<u>ORDER_ DOLLARS</u>
Apple Inc.	1000

Historically accurate

```
SELECT
  company_name_historic,
  sum(order_dollars)
FROM
  :
```



<u>COMPANY_ NAME_HISTORIC</u>	<u>ORDER_ DOLLARS</u>
Apple Computer, Inc.	600
Apple Inc.	400

Type 1 / 2 Hybrid

Current info

```
SELECT
  company_name_current,
  sum(order_dollars)
FROM
  :
```

Historically accurate

```
SELECT
  company_name_historic,
  sum(order_dollars)
FROM
  :
```

How would we do the above if we only had a type 2 dimension?

Type 1 / 2 / 3 Hybrid

3 columns for an attribute

current value, previous value, historic value