Analytic views in Oracle 12.2

The virtual cube

Kim Berg Hansen
Senior Consultant



About me



- Danish geek
- SQL & PL/SQL developer since 2000
- Developer at Trivadis AG since 2016
  http://www.trivadis.dk
- Oracle Certified Expert in SQL
- Oracle ACE
- Blogger at http://www.kibeha.dk
- SQL quizmaster at
  http://devgym.oracle.com
- Likes to cook
- Reads sci-fi
- Member of Danish Beer Enthusiasts

trivadis
makes IT easier.

500+ Technical Experts Helping Peers Globally

ORACLE ACE PROGRAM







---

3 Membership Tiers

- Oracle ACE Director
- Oracle ACE
- Oracle ACE Associate

bit.ly/OracleACEProgram

Connect:

oracle-ace_ww@oracle.com

Facebook.com/oracleaces

@oracleace

ORACLE Developer Community

Nominate yourself or someone you know: acenomination.oracle.com

About Trivadis

Trivadis is a market leader in IT consulting, system integration, solution engineering and the provision of IT services focusing on ORACLE and Microsoft technologies in Switzerland, Germany, Austria and Denmark.
We offer our services in the following strategic business fields:



Trivadis Services takes over the interacting operation of your IT systems.

trivadis makes IT easier.

With over 600 specialists and IT experts in your region



- 14 Trivadis branches and more than 600 employees
- 260 Service Level Agreements
- Over 4,000 training participants
- Research and development budget: EUR 5.0 million
- Financially self-supporting and sustainably profitable
- Experience from more than 1,900 projects per year at over 800 customers

trivadis
makes IT easier.

Analytic Views in Oracle 12.2 - Agenda

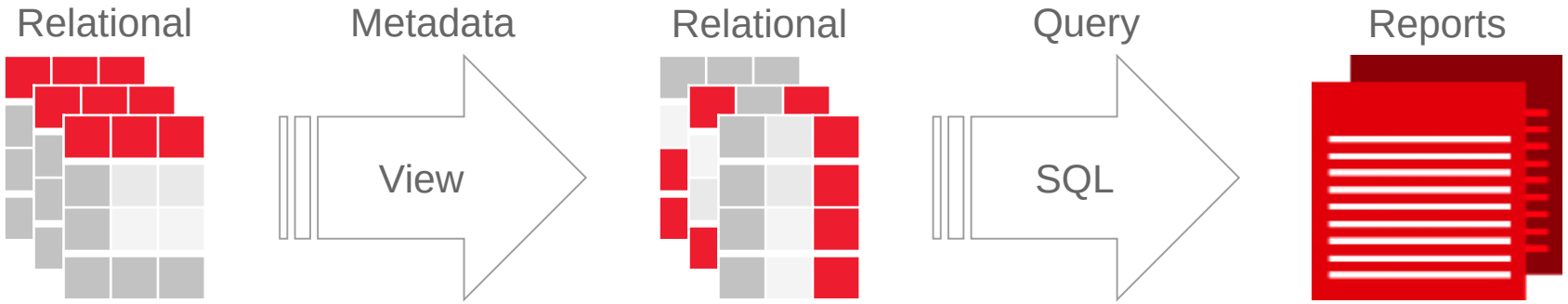
1. Concepts
2. ATTRIBUTE DIMENSION
3. HIERARCHY
4. ANALYTIC VIEW
5. Querying with SQL
6. Performance
7. Classifications
8. Summary

trivadis makes IT easier.

Concepts

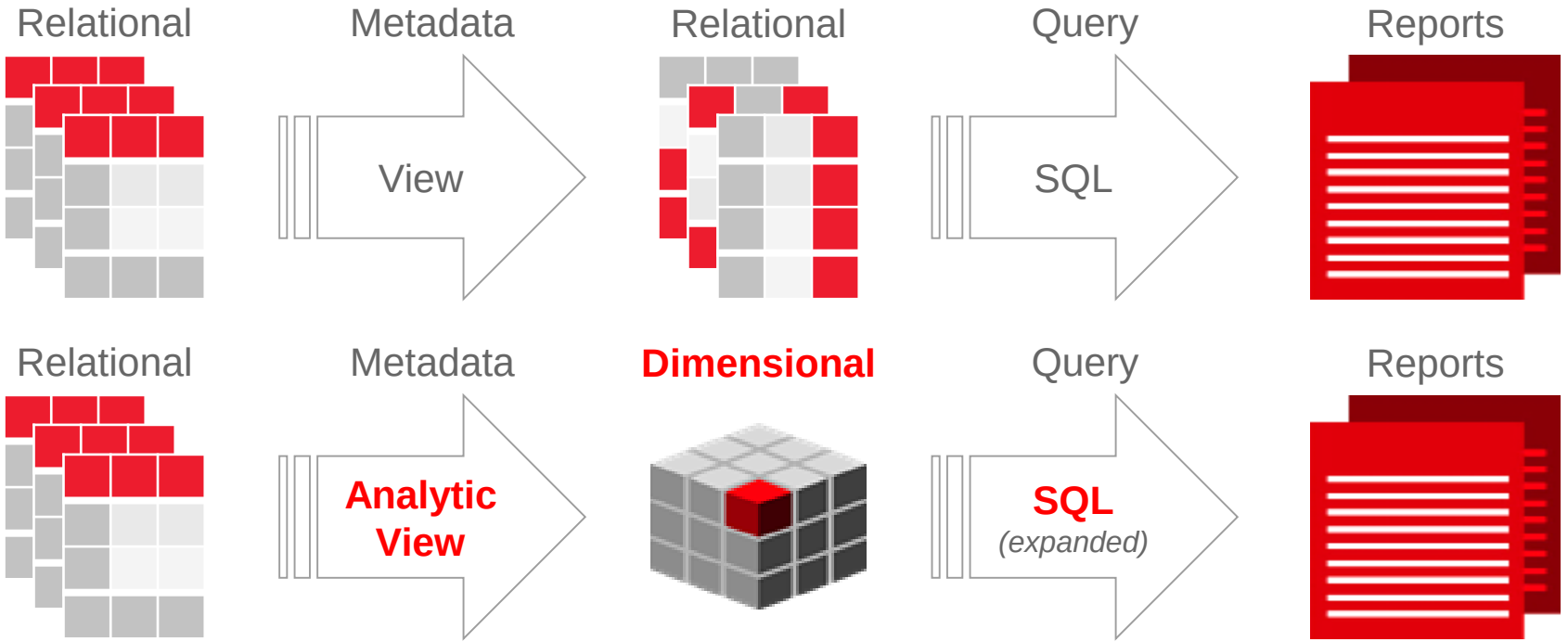
trivadis
makes IT easier.

Like a view – but dimensional



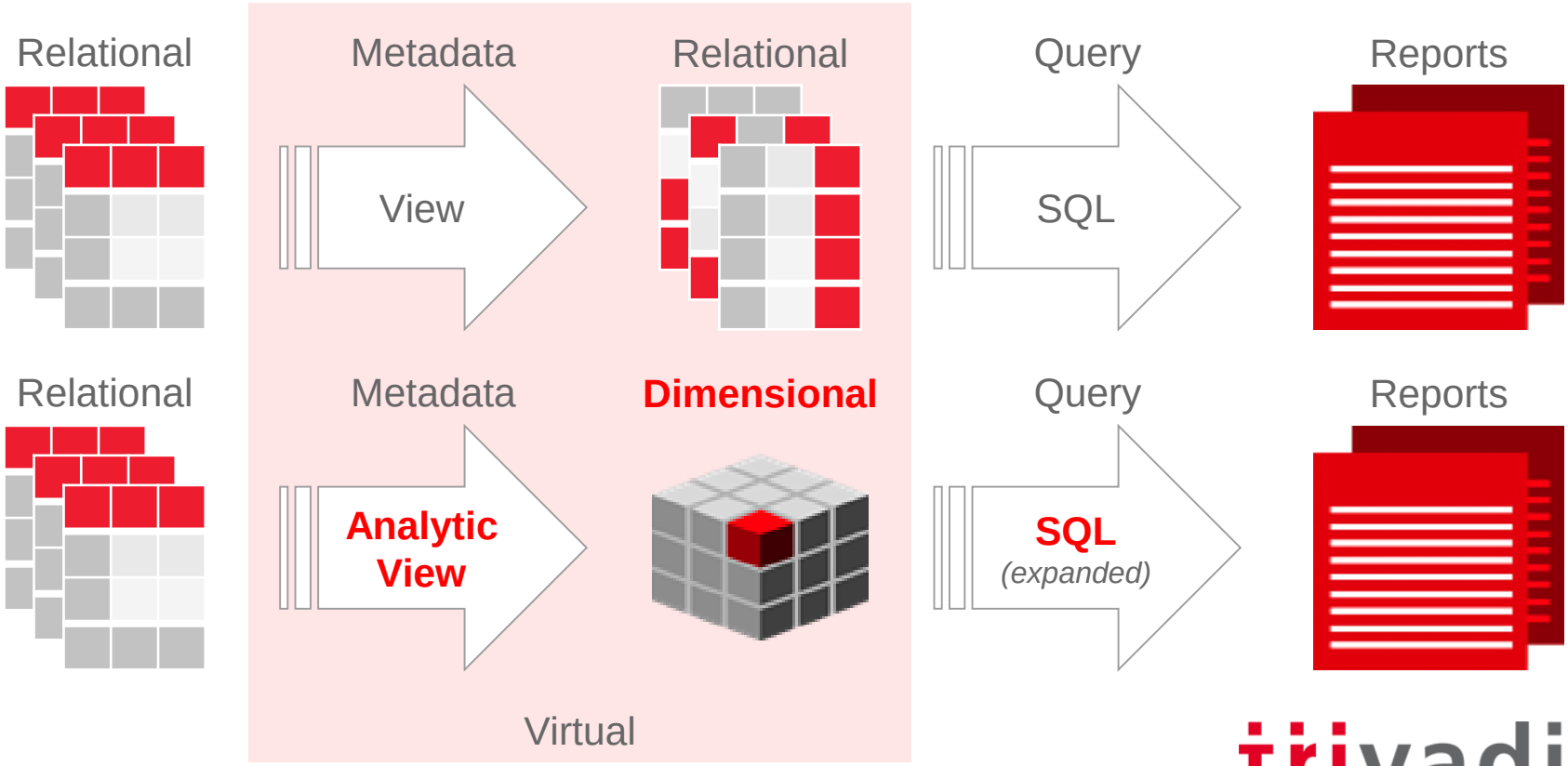
trivadis
makes IT easier.

Like a view – but dimensional



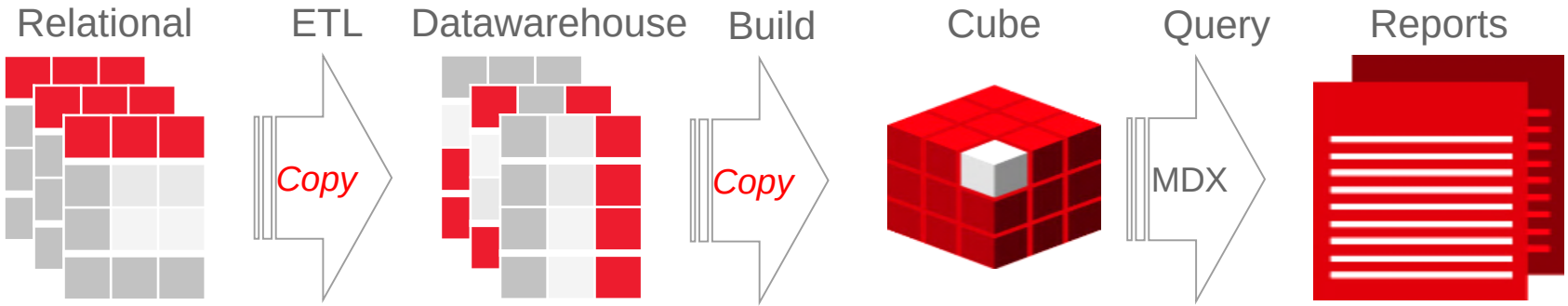
trivadis
makes IT easier.

Like a view – but dimensional



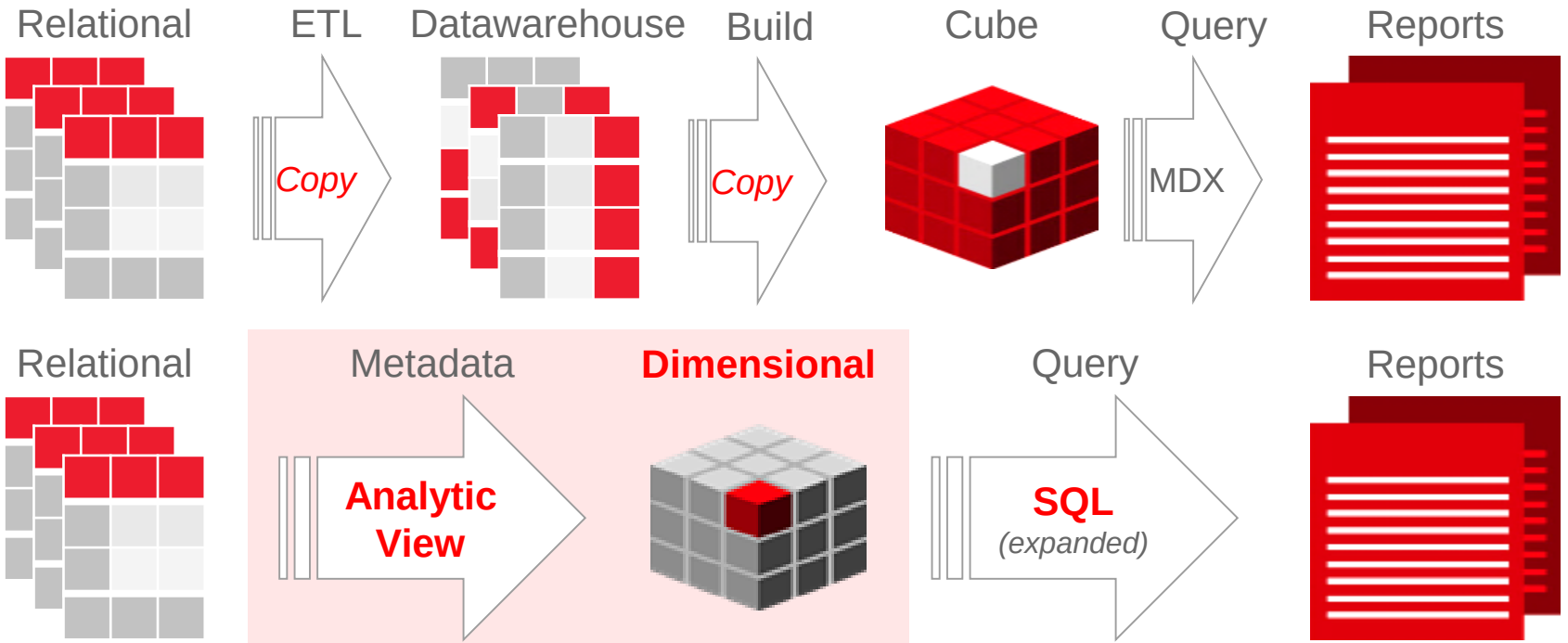
trivadis
makes IT easier.

Like a dimensional OLAP cube – but virtual



trivadis
makes IT easier.

Like a dimensional OLAP cube – but virtual



trivadis
makes IT easier.

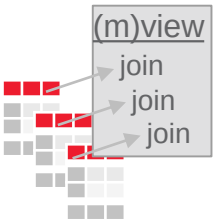
Anatomy of Analytic View: Metadata object types

Dimension Table



Tables+view

(*Materialized?*)



Relational

Fact Table



Relational

trivadis makes IT easier.

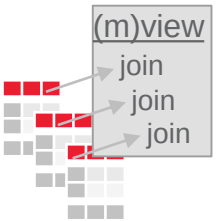
Anatomy of Analytic View: Metadata object types

Dimension Table



Tables+view

(*Materialized?*)



Relational

Attribute Dimension

Attr. Dimension

Definition of:

- Attributes
- Levels
- Member keys
- Member names

Attr. Dimension

Definition of:

- Attributes
- Levels
- Member keys
- Member names

Virtual (*not copying data*)

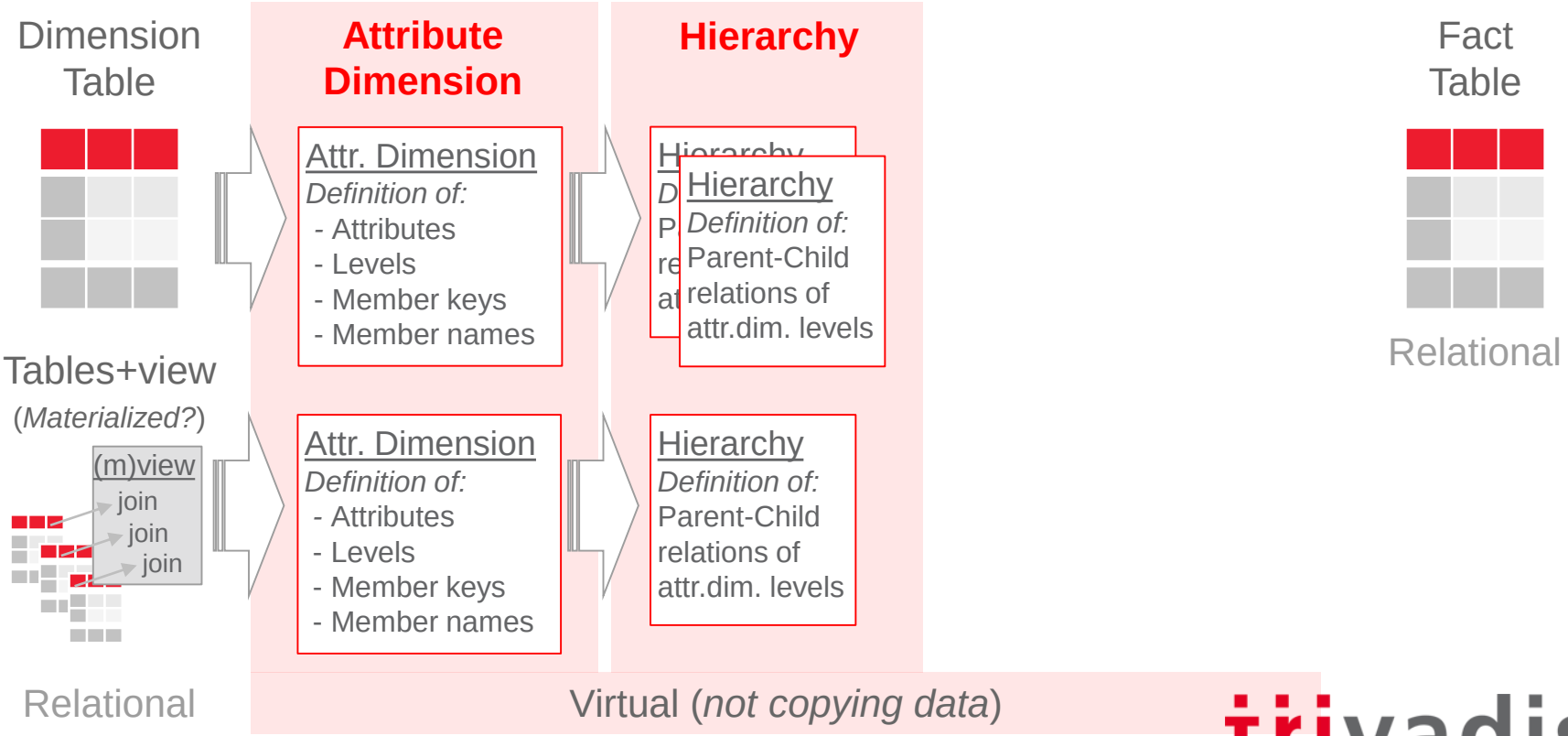
Fact Table



Relational

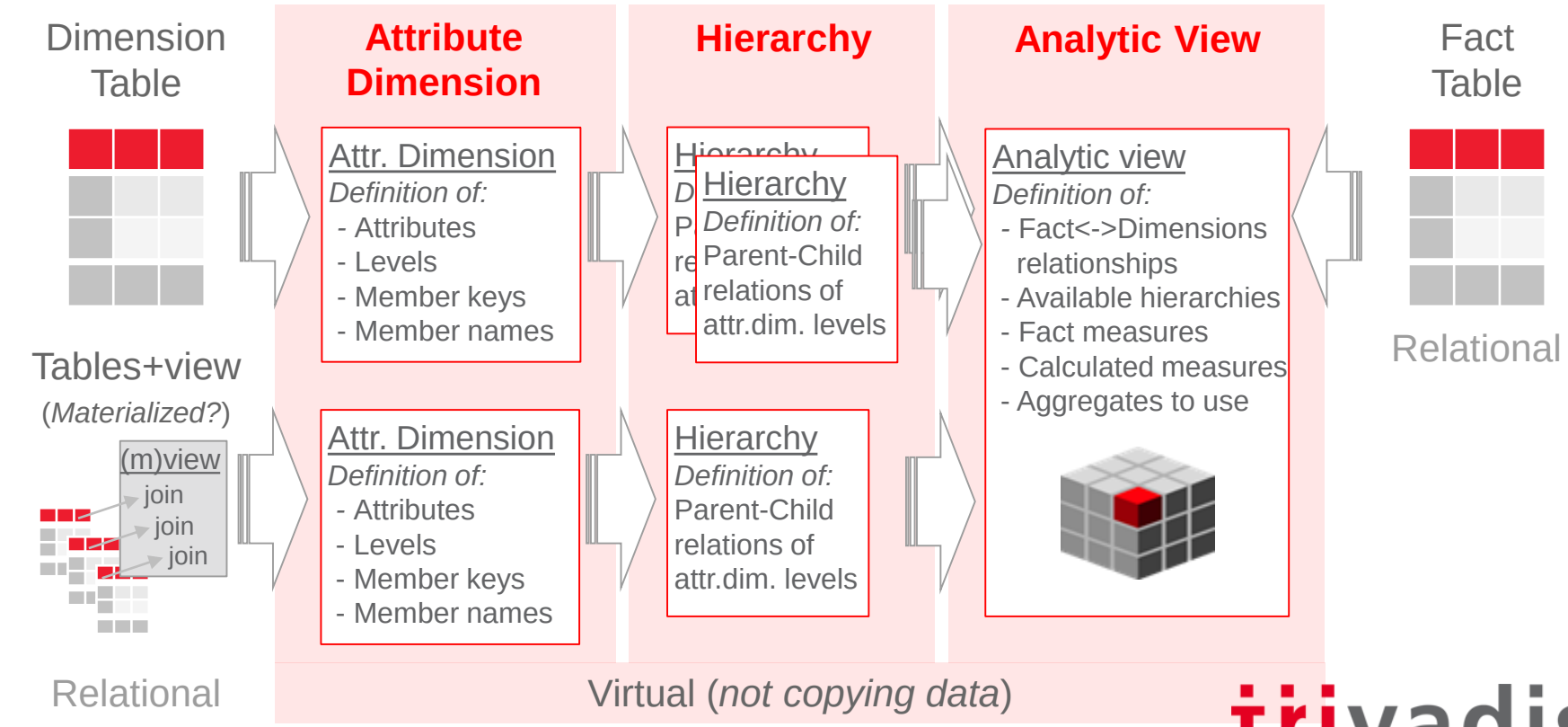
trivadis makes IT easier.

Anatomy of Analytic View: Metadata object types



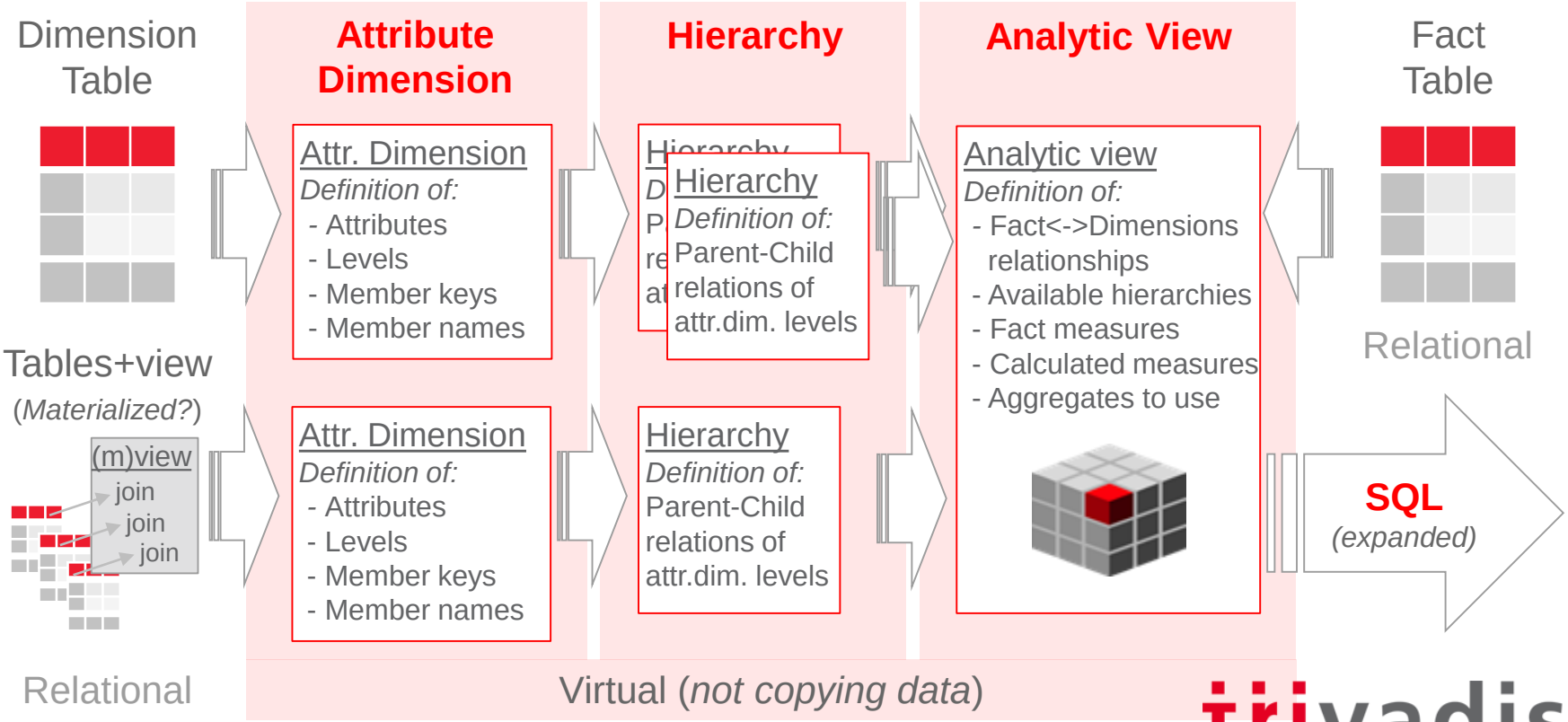
trivadis makes IT easier.

Anatomy of Analytic View: Metadata object types



trivadis makes IT easier.

Anatomy of Analytic View: Metadata object types



trivadis makes IT easier.

12.2 Application Containers

- SHARE WITH clause is available for the object types
  - Attribute dimensions
  - Hierarchies
  - Analytic views
- Allows the objects to be shared across application containers
- Sharing can be either
  - NONE - share nothing (default)
  - METADATA - share metadata (definition) only
  - OBJECT - share entire object, both metadata as well as data

trivadis makes IT easier.

ATTRIBUTE DIMENSION

trivadis
makes IT easier.

ATTRIBUTE DIMENSION

- Metadata – not storing values. Values stay in base table(s)
- Based on a regular table with dimension values
- Or multiple tables joined and denormalized:
  - via regular view
  - or via materialized view
- Defining all possible member attributes of the dimension
- Defining all possible levels of the dimension:
  - which attributes are keys, alternate keys, names and ordering
  - which attributes determine other attributes

trivadis makes IT easier.

ATTRIBUTE DIMENSION - Standard

Denormalized dimension table

```
create table item_dim (
   item_id         integer       primary key
 , item_name       varchar2(20)  not null unique
 , category_id     integer       not null
 , category_name   varchar2(20)  not null
);
```


trivadis makes IT easier.

ATTRIBUTE DIMENSION - Standard

Denormalized dimension table

```
create table item_dim (
   item_id
 , item_nam
 , category
 , category
);
```

Attribute dimension

```
create or replace attribute dimension item_attr_dim
   dimension type standard
   using          item_dim
attributes (
   item_id, item_name, category_id, category_name
)
level item        key             item_id
                  member name     item_name
                  order by        item_name
                  determines      (category_id)
level category    key             category_id
                  member name     category_name
                  order by        category_name
all               member name     'ALL ITEMS';
```


trivadis makes IT easier.

ATTRIBUTE DIMENSION – Branched levels

Normalized dimension tables

```
create table country_dim (
   country_id      integer       primary key
 , country_name    varchar2(20)  not null     );
create table state_dim (
   state_id        integer       primary key
 , state_name      varchar2(20)  not null
 , country_id      integer       not null     references country_dim    );
create table custgroup_dim (
   custgroup_id    integer       primary key
 , custgroup_name  varchar2(20)  not null     );
create table customer_dim (
   customer_id     integer       primary key
 , customer_name   varchar2(20)  not null
 , state_id        integer       not null     references state_dim
 , custgroup_id    integer       not null     references custgroup_dim );
```


trivadis makes IT easier.

ATTRIBUTE DIMENSION – Branched levels

Normalized dimension tables

Denormalized dimension view

```
create or replace view customer_dim_view as
select c.customer_id      , c.customer_name
     , cg.custgroup_id    , cg.custgroup_name
     , s.state_id         , s.state_name
     , co.country_id      , co.country_name
  from customer_dim  c
  join custgroup_dim cg  on cg.custgroup_id = c.custgroup_id
  join state_dim     s   on s.state_id = c.state_id
  join country_dim   co  on co.country_id = s.country_id;


/*
Could be materialized view - periodic or ON COMMIT refresh
*/
```


trivadis makes IT easier.

ATTRIBUTE DIMENSION – Branched levels

Normalized dimension tables

Denormalized dimension view

Attribute dimension

```
create table country_dim (
   co
 , co
creat
   st
 , st
 , co
creat
   cu
 , cu
creat
   cu
 , cu
 , st
 , cu
```

```
creat
selec
      fro
      joi
      joi
      joi

/*
Could
*/
```

```
create or replace attribute dimension customer_attr_dim
using customer_dim_view
attributes ( customer_id, customer_name, custgroup_id, custgroup_name
           , state_id   , state_name   , country_id  , country_name  )
level customer    key           customer_id
                  member name   customer_name
                  determines    (custgroup_id, state_id)
level custgroup   key           custgroup_id
                  member name   custgroup_name
level state       key           state_id
                  member name   state_name
                  determines    (country_id)
level country     key           country_id
                  member name   country_name
all               member name   'ALL CUSTOMERS';
```


trivadis makes IT easier.

ATTRIBUTE DIMENSION – Dimension type TIME

Dimension table

```
create table date_dim (
   date_id      date            primary key
 , date_name    varchar2(10)    not null unique
 , month_id     integer         not null
 , month_name   varchar2(10)    not null
 , qtr_id       integer         not null
 , qtr_name     varchar2(10)    not null
 , year_id      integer         not null
 , year_name    varchar2(4)     not null
 , mthofyr_id   integer         not null
 , mthofyr_name varchar2(3)     not null
 , season_id    integer         not null
 , season_name varchar2(10)     not null
);
```


trivadis makes IT easier.

ATTRIBUTE DIMENSION – Dimension type TIME

Dimension table

```
create table date_dim (
)
```

Populate table

```
insert into date_dim
select date_date as date_id                                  , to_char(date_date,'YYYY-MM-DD') date_name
     , to_number(to_char(date_date,'YYYYMM')) month_id, to_char(date_date,'YYYY-MM') month_name
     , to_number(to_char(date_date,'YYYYQ')) qtr_id    , to_char(date_date,'YYYY-"Q"Q') qtr_name
     , to_number(to_char(date_date,'YYYY')) year_id    , to_char(date_date,'YYYY') year_name
     , extract(month from date_date) mthofyr_id        , to_char(date_date,'FMMon'
                                                                 , 'nls_date_language=american') mthofyr_name
     , case when extract(month from date_date) in (12,1,2)  then 1
            when extract(month from date_date) in (3,4,5 )  then 2
            when extract(month from date_date) in (6,7,8 )  then 3
            when extract(month from date_date) in (9,10,11) then 4        end season_id
     , case when extract(month from date_date) in (12,1,2)  then 'Winter'
            when extract(month from date_date) in (3,4,5 )  then 'Spring'
            when extract(month from date_date) in (6,7,8 )  then 'Summer'
            when extract(month from date_date) in (9,10,11) then 'Fall'   end season_name
  from (
   select date '2014-01-01' + level - 1 date_date  from dual
   connect by level <= date '2017-12-31' - date '2014-01-01' + 1
  )
 order by date_id;
```


trivadis makes IT easier.

ATTRIBUTE DIMENSION – Dimension type TIME

Dimension table

Populate table

Attribute dimension

```
create table date_dim (
...
)

insert into date_dim
s

create or replace attribute dimension date_attr_dim  dimension type time  using date_dim
attributes ( date_id , date_name , month_id   , month_name   , qtr_id    , qtr_name
           , year_id , year_name , mthofyr_id , mthofyr_name , season_id , season_name  )
level day      level type  days      key         date_id      member name date_name
               order by    date_id   determines  (month_id, mthofyr_id)
level month    level type  months    key         month_id     member name month_name
               order by    month_id  determines  (qtr_id, mthofyr_id)
level quarter  level type  quarters  key         qtr_id       member name qtr_name
               order by    qtr_id    determines  (year_id)
level year     level type  years     key         year_id      member name year_name
               order by    year_id
level mthofyr  level type  months    key         mthofyr_id   member name mthofyr_name
               order by    mthofyr_id determines  (season_id)
level season   level type  quarters  key         season_id    member name season_name
               order by    season_id
all            member name 'ALL DATES';
```


trivadis makes IT easier.

HIERARCHY

trivadis
makes IT easier.

HIERARCHY

- Metadata – not storing values. Values stay in base table(s)
- Based on attribute dimension
- Defining parent->child relations for the levels of a dimension
- Multiple hierarchies may exist for one dimension
- Generates virtual columns and rows for the hierarchy
- Is a row source – you can SELECT from a hierarchy

trivadis makes IT easier.

HIERARCHY - Single

Attribute dimension

```
create or replace attribute dimension item_attr_dim
   dimension type standard
   using          item_dim
attributes (
   item_id, item_name, category_id, category_name
)
level item        key             item_id
                  member name     item_name
                  order by        item_name
                  determines      (category_id)
level category    key             category_id
                  member name     category_name
                  order by        category_name
all               member name     'ALL ITEMS';
```


trivadis makes IT easier.

HIERARCHY - Single

Attribute dimension

```
create or replace attribute dimension item_attr_dim
   dimensio
   using
attributes
   item_id,
)
level item


level categ


all
```

Hierarchy

```
create or replace hierarchy item_category_hier
using item_attr_dim
(
   item
   child of category
);
```


trivadis makes IT easier.

Columns of a hierarchy

Dimension attribute columns / rows + generated hierarchical columns / rows

```
ITEM_ID
ITEM_NAME
CATEGORY_ID
CATEGORY_NAME
MEMBER_NAME
MEMBER_UNIQUE_NAME
MEMBER_CAPTION
MEMBER_DESCRIPTION
LEVEL_NAME
HIER_ORDER
DEPTH
IS_LEAF
PARENT_LEVEL_NAME
PARENT_UNIQUE_NAME
```


trivadis makes IT easier.

Columns of a hierarchy

Dimension attribute columns / rows + generated hierarchical columns / rows

ITEM_ID	230
ITEM_NAME	Bikini
CATEGORY_ID	2
CATEGORY_NAME	For the beach
MEMBER_NAME	Bikini
MEMBER_UNIQUE_NAME	[ITEM].&[230]
MEMBER_CAPTION	
MEMBER_DESCRIPTION	
LEVEL_NAME	ITEM
HIER_ORDER	2
DEPTH	2
IS_LEAF	1
PARENT_LEVEL_NAME	CATEGORY
PARENT_UNIQUE_NAME	[CATEGORY].&[2]

trivadis makes IT easier.

Columns of a hierarchy

Dimension attribute columns / rows + generated hierarchical columns / rows

ITEM_ID	230	
ITEM_NAME	Bikini	
CATEGORY_ID	2	2
CATEGORY_NAME	For the b	For the beach
MEMBER_NAME	Bikini	For the beach
MEMBER_UNIQUE_NAME	[ITEM].&[	[CATEGORY].&[2]
MEMBER_CAPTION		
MEMBER_DESCRIPTION		
LEVEL_NAME	ITEM	CATEGORY
HIER_ORDER	2	1
DEPTH	2	1
IS_LEAF	1	0
PARENT_LEVEL_NAME	CATEGORY	ALL
PARENT_UNIQUE_NAME	[CATEGORY	[ALL].[ALL ITEMS]

trivadis makes IT easier.

Columns of a hierarchy

Dimension attribute columns / rows + generated hierarchical columns / rows

ITEM_ID	230		130
ITEM_NAME	Bikini		Bob sled
CATEGORY_ID	2	2	1
CATEGORY_NAME	For the b	For the bea	Winter sports
MEMBER_NAME	Bikini	For the bea	Bob sled
MEMBER_UNIQUE_NAME	[ITEM].&[	[CATEGORY]	[ITEM].&[130]
MEMBER_CAPTION			
MEMBER_DESCRIPTION			
LEVEL_NAME	ITEM	CATEGORY	ITEM
HIER_ORDER	2	1	7
DEPTH	2	1	2
IS_LEAF	1	0	1
PARENT_LEVEL_NAME	CATEGORY	ALL	CATEGORY
PARENT_UNIQUE_NAME	[CATEGORY	[ALL].[ALL	[CATEGORY].&[1]

trivadis makes IT easier.

Columns of a hierarchy

Dimension attribute columns / rows + generated hierarchical columns / rows

ITEM_ID	230		130	
ITEM_NAME	Bikini		Bob sled	
CATEGORY_ID	2	2	1	1
CATEGORY_NAME	For the b	For the bea	Winter sp	Winter sports
MEMBER_NAME	Bikini	For the bea	Bob sled	Winter sports
MEMBER_UNIQUE_NAME	[ITEM].&[	[CATEGORY]	[ITEM].&[	[CATEGORY].&[1]
MEMBER_CAPTION				
MEMBER_DESCRIPTION				
LEVEL_NAME	ITEM	CATEGORY	ITEM	CATEGORY
HIER_ORDER	2	1	7	6
DEPTH	2	1	2	1
IS_LEAF	1	0	1	0
PARENT_LEVEL_NAME	CATEGORY	ALL	CATEGORY	ALL
PARENT_UNIQUE_NAME	[CATEGORY	[ALL].[ALL	[CATEGORY	[ALL].[ALL ITEMS]

trivadis makes IT easier.

Columns of a hierarchy

Dimension attribute columns / rows + generated hierarchical columns / rows

ITEM_ID	230		130		
ITEM_NAME	Bikini		Bob sled		
CATEGORY_ID	2	2	1	1	
CATEGORY_NAME	For the b	For the bea	Winter sp	Winter spo	
MEMBER_NAME	Bikini	For the bea	Bob sled	Winter spo	ALL ITEMS
MEMBER_UNIQUE_NAME	[ITEM].&[	[CATEGORY]	[ITEM].&[	[CATEGORY]	[ALL].[ALL ITEMS]
MEMBER_CAPTION					
MEMBER_DESCRIPTION					
LEVEL_NAME	ITEM	CATEGORY	ITEM	CATEGORY	ALL
HIER_ORDER	2	1	7	6	0
DEPTH	2	1	2	1	0
IS_LEAF	1	0	1	0	0
PARENT_LEVEL_NAME	CATEGORY	ALL	CATEGORY	ALL	
PARENT_UNIQUE_NAME	[CATEGORY	[ALL].[ALL	[CATEGORY	[ALL].[ALL	

trivadis makes IT easier.

Hierarchy tree

HIER_ORDER gives tree ordering, use DEPTH and IS_LEAF to picture tree

```
select hier_order
     , lpad(' ', depth * 2)
       || case is_leaf
            when 0 then '+ '
                   else '- '
          end
       || member_name as tree
     , member_unique_name
  from item_category_hier
 order by hier_order;
```


trivadis makes IT easier.

Hierarchy tree

HIER_ORDER gives tree ordering, use DEPTH and IS_LEAF to picture tree

```
select hier_order
     , lpad(' ', depth * 2)
       || case is_leaf
            when 0 then '+ '
                   else '- '
          end
       || member_name as tree
     , member_unique_name
  from item_category_hier
 order by hier_order;
```

```
HIER_ORDER TREE                 MEMBER_UNIQUE_NAME
---------- -------------------- ------------------
         0 + ALL ITEMS          [ALL].[ALL ITEMS]
         1   + For the beach    [CATEGORY].&[2]
         2     - Bikini         [ITEM].&[230]
         3     - Sunglasses     [ITEM].&[200]
         4     - Surfboard      [ITEM].&[220]
         5     - Volleyball     [ITEM].&[210]
         6   + Winter sports    [CATEGORY].&[1]
         7     - Bob sled       [ITEM].&[130]
         8     - Ice skates     [ITEM].&[110]
         9     - Ski gloves     [ITEM].&[120]
        10     - Snowboard      [ITEM].&[100]

11 rows selected.
```


trivadis makes IT easier.

HIERARCHY - Multiple

Attribute dimension

```
create or replace attribute dimension customer_attr_dim
using customer_dim_view
attributes ( customer_id, customer_name, custgroup_id, custgroup_name
           , state_id   , state_name   , country_id  , country_name  )
level customer    key             customer_id
                  member name     customer_name
                  determines      (custgroup_id, state_id)
level custgroup   key             custgroup_id
                  member name     custgroup_name
level state       key             state_id
                  member name     state_name
                  determines      (country_id)
level country     key             country_id
                  member name     country_name
all               member name     'ALL CUSTOMERS';
```


trivadis makes IT easier.

HIERARCHY - Multiple

Attribute dimension

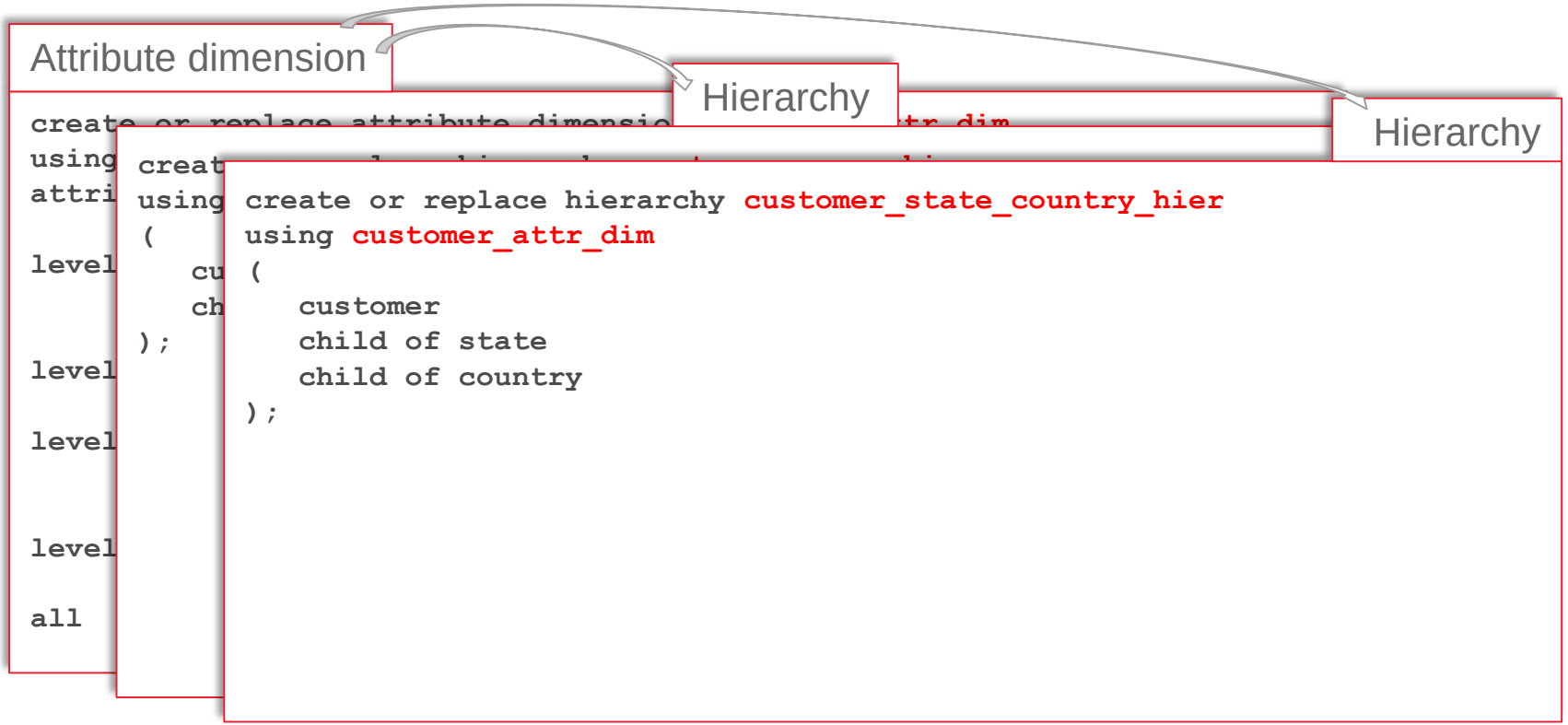
```
create or replace attribute dimensio
using
attri

level

level

level

level

all
```

Hierarchy

```
create or replace hierarchy customer_group_hier
using customer_attr_dim
(
   customer
   child of custgroup
);
```


trivadis makes IT easier.

HIERARCHY - Multiple



trivadis makes IT easier.

HIERARCHY – Dimension type TIME

Attribute dimension

```
create or replace attribute dimension date_attr_dim  dimension type time  using date_dim
attributes ( date_id , date_name , month_id    , month_name    , qtr_id     , qtr_name
           , year_id , year_name , mthofyr_id , mthofyr_name , season_id , season_name  )
level day       level type  days          key          date_id      member name date_name
                order by    date_id       determines   (month_id, mthofyr_id)
level month     level type  months        key          month_id     member name month_name
                order by    month_id      determines   (qtr_id, mthofyr_id)
level quarter   level type  quarters      key          qtr_id       member name qtr_name
                order by    qtr_id        determines   (year_id)
level year      level type  years         key          year_id      member name year_name
                order by    year_id
level mthofyr   level type  months        key          mthofyr_id   member name mthofyr_name
                order by    mthofyr_id    determines   (season_id)
level season    level type  quarters      key          season_id    member name season_name
                order by    season_id
all             member name 'ALL DATES';
```


trivadis makes IT easier.

HIERARCHY – Dimension type TIME

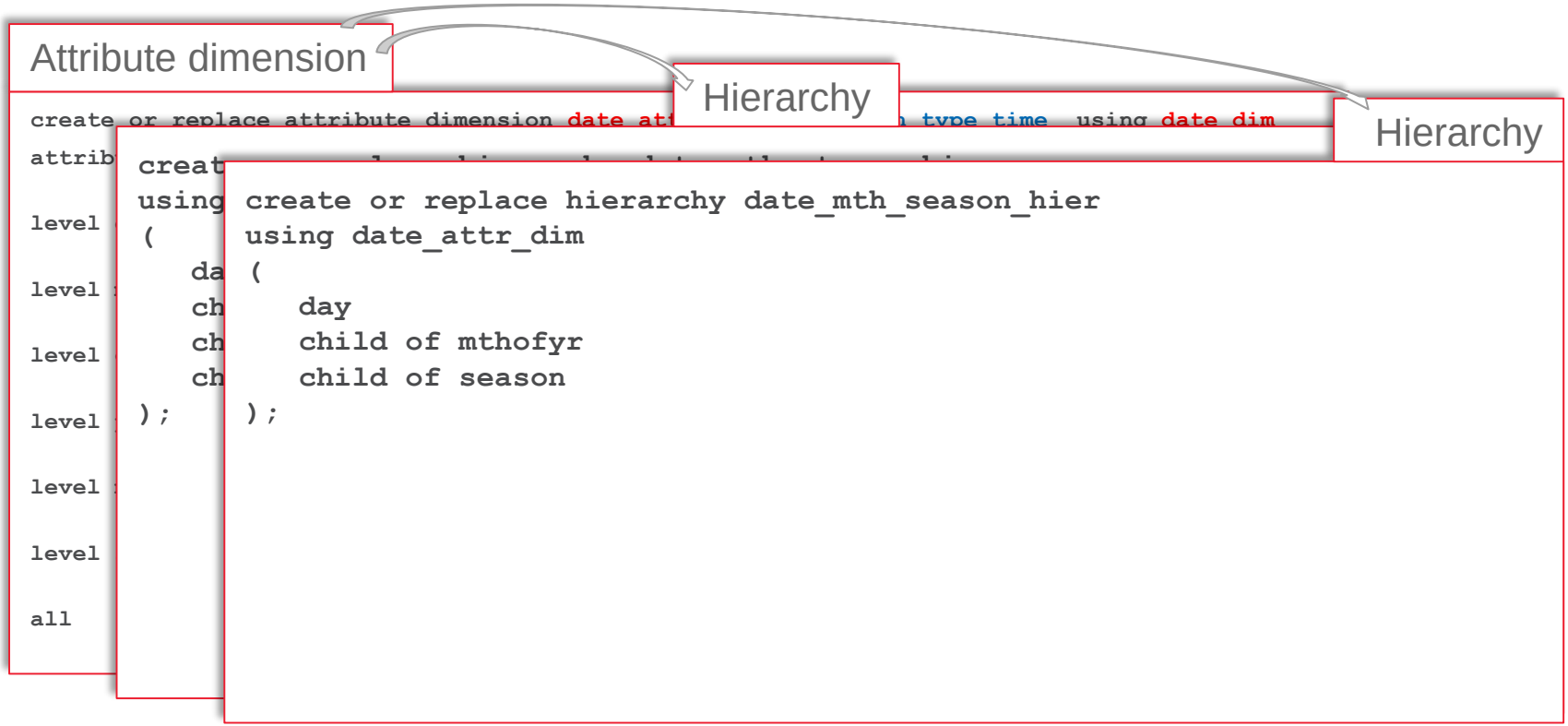
Attribute dimension

Hierarchy

```
create or replace hierarchy date_mth_qtr_yr_hier
using date_attr_dim
(
   day
   child of month
   child of quarter
   child of year
);
```


trivadis makes IT easier.

HIERARCHY – Dimension type TIME



trivadis makes IT easier.

ANALYTIC VIEW

trivadis
makes IT easier.

ANALYTIC VIEW

- Metadata – not storing values. Values stay in base table
- Based on fact table, attribute dimensions and hierarchies
- Defining relations between fact and one or more dimensions
- Defining which hierarchies of the dimension can be used in the view
- Defining measures:
  - Based on fact columns or calculations
  - Expanded analytic syntax supports hierarchies
  - Defines which aggregations to be used on measures

trivadis makes IT easier.

ANALYTIC VIEW

Fact table

```
create table sales_fact (
   sales_date  date     not null references date_dim
 , item_id     integer  not null references item_dim
 , customer_id integer  not null references customer_dim
 , qty         number   not null
 , amount      number   not null
 , cost        number   not null
);
```


trivadis makes IT easier.

ANALYTIC VIEW

Fact table

```
create table sales_fact (
   sales_da
 , item_id
 , customer
 , qty
 , amount
 , cost
);
```

Analytic view

```
create or replace analytic view sales_av
using sales_fact
dimension by (

    ((dimension clauses))

)
measures (

    ((measure clauses))

)
default measure      amount
default aggregate by sum;
```


trivadis makes IT easier.

Dimension clauses

Analytic view

```
create or replace analytic view sales_av
using sales_fact
dimension by (

    ((dimension clauses))

)
measures (

    ((measure clauses))

)
default measure      amount
default aggregate by sum;
```


trivadis makes IT easier.

Dimension clauses

Analytic view

```
create or replace analytic view sales_av
using sales
dimension b
    ((dimens
)
measures (
    ((measur
)
default mea
default agg
```

Dimension clauses

```
   date_attr_dim
      key  sales_date   references  date_id
      hierarchies ( date_mth_qtr_yr_hier    default
                  , date_wk_yr_hier
                  , date_mth_season_hier            )
 , item_attr_dim
      key  item_id      references  item_id
      hierarchies ( item_category_hier      default )
 , customer_attr_dim
      key  customer_id  references  customer_id
      hierarchies ( customer_group_hier     default
                  , customer_state_country_hier     )
```


trivadis makes IT easier.

Measure clauses

Analytic view

```
create or replace analytic view sales_av
using sales_fact
dimension by (

    ((dimension clauses))

)
measures (

    ((measure clauses))

)
default measure      amount
default aggregate by sum;
```


trivadis makes IT easier.

Measure clauses

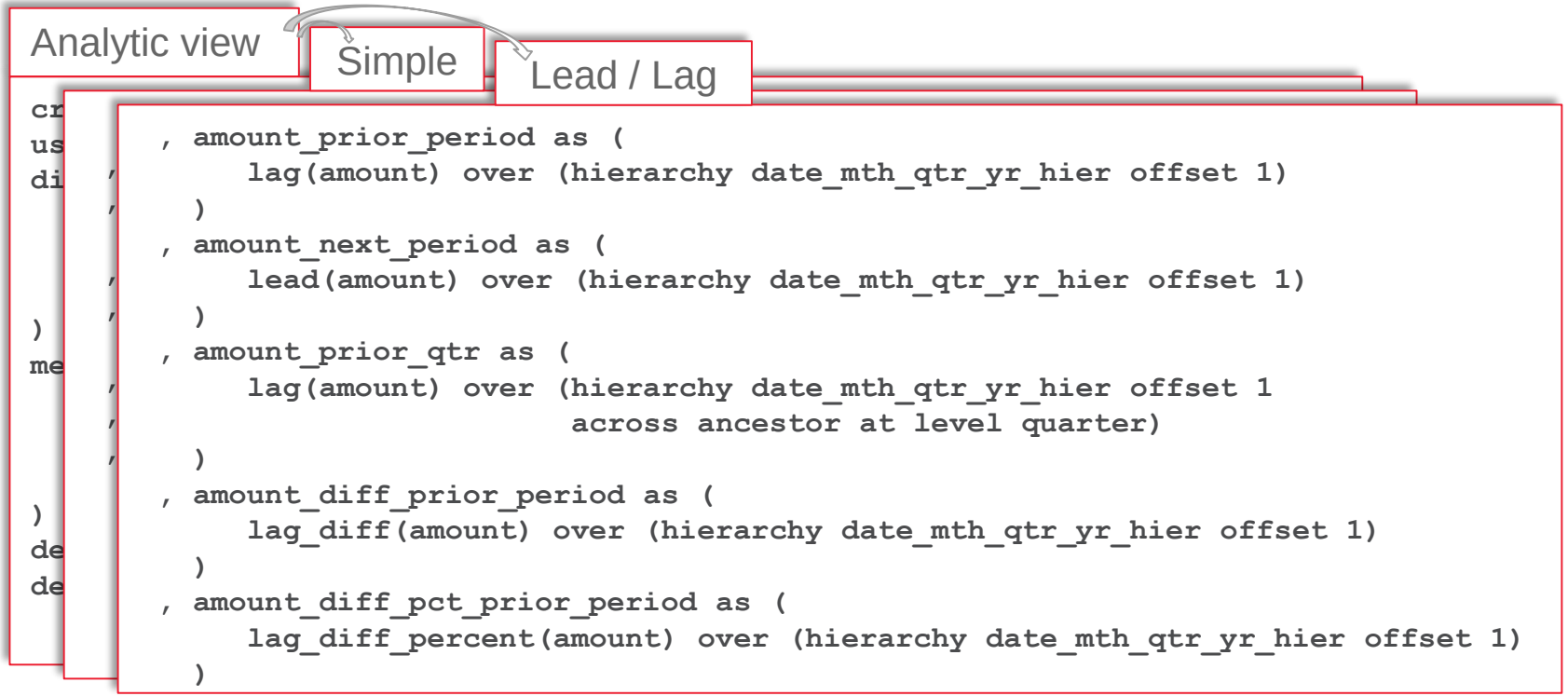
Analytic view

Simple

```
cr
us
di
    qty           fact  qty
  , amount        fact  amount
  , cost          fact  cost

  , max_qty       fact  qty    aggregate by max
  , avg_amount    fact  amount aggregate by avg
)
me
  , profit        as    (amount - cost)
  , piece_profit as     (profit / nullif(qty,0))
  , margin        as    (100 * profit / nullif(amount,0))

)
de
de
```

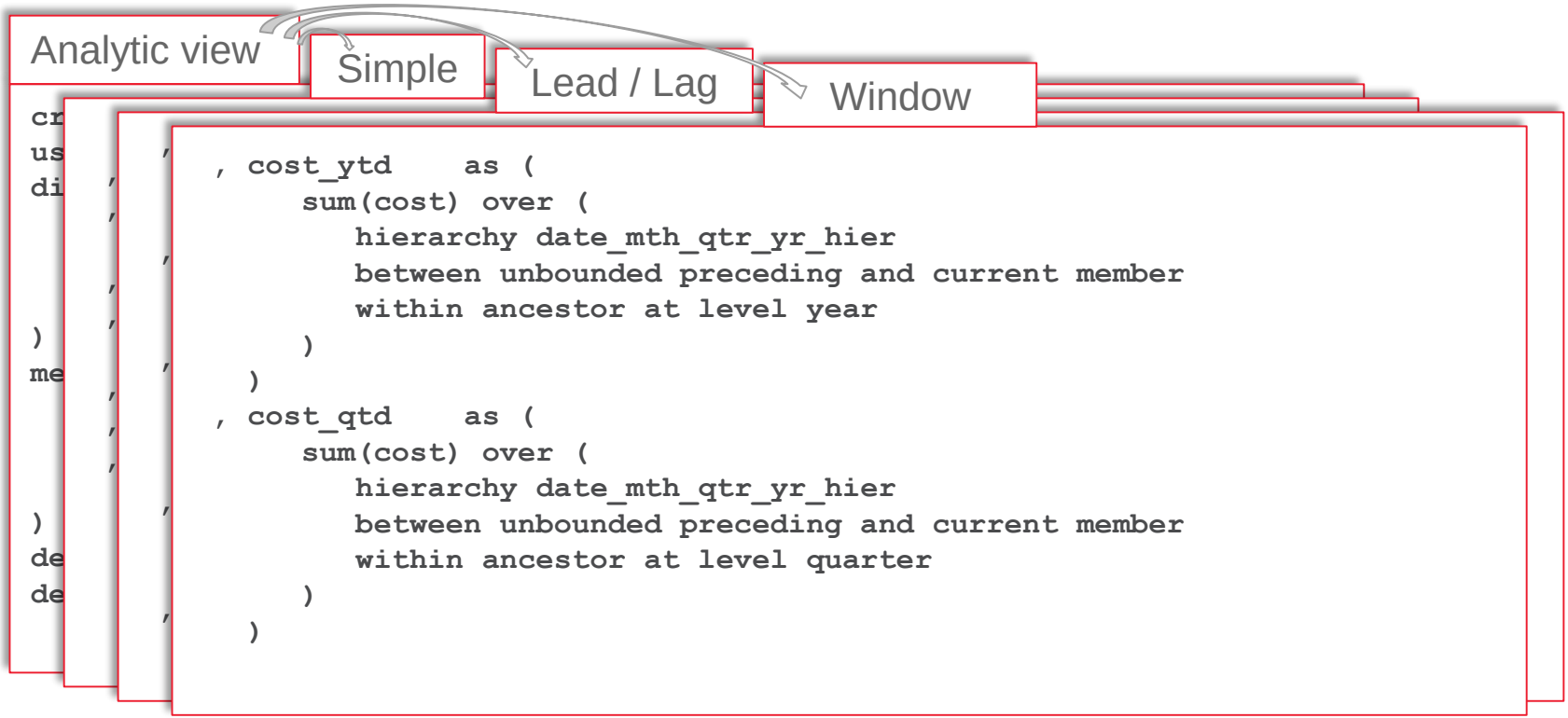

trivadis makes IT easier.

Measure clauses



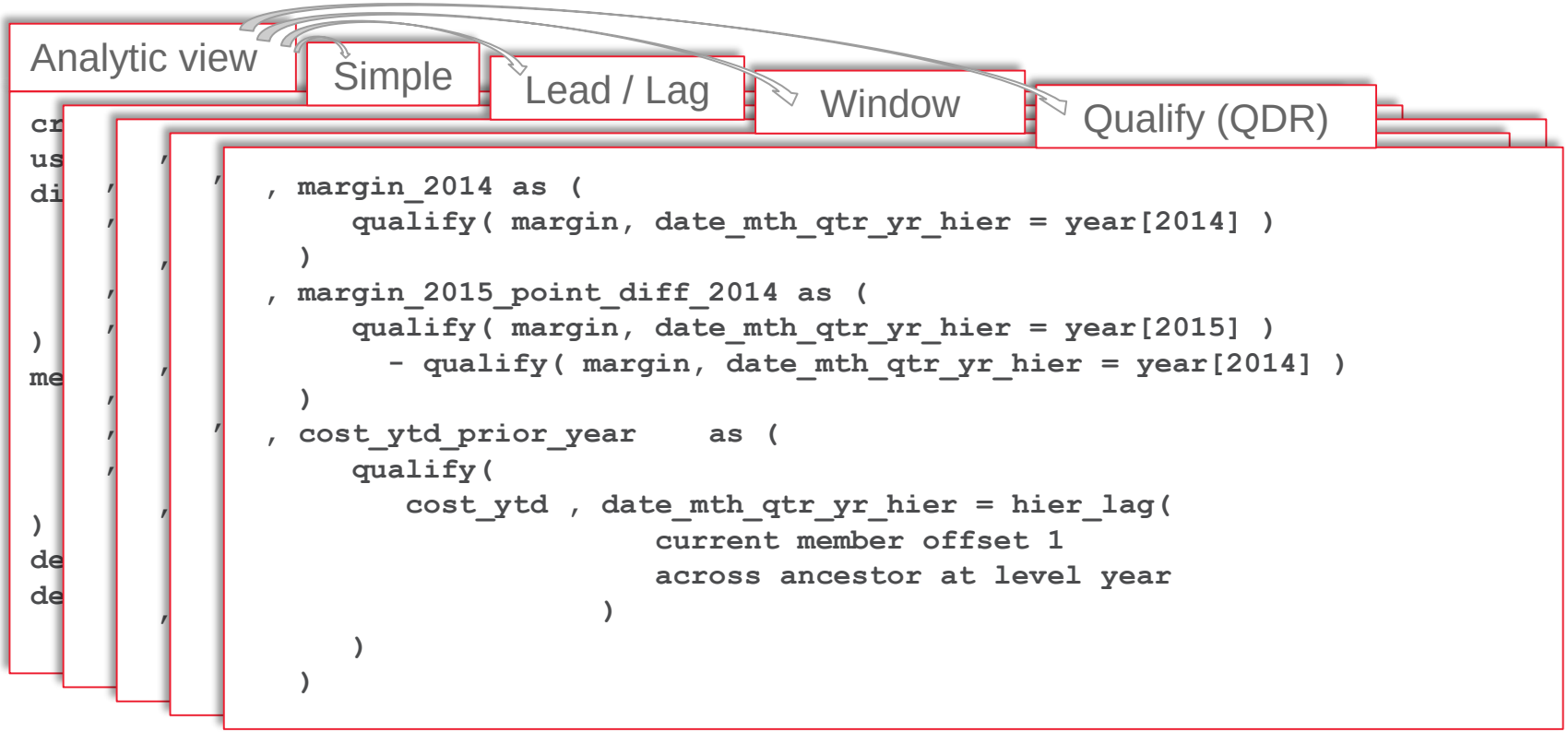
trivadis makes IT easier.

Measure clauses



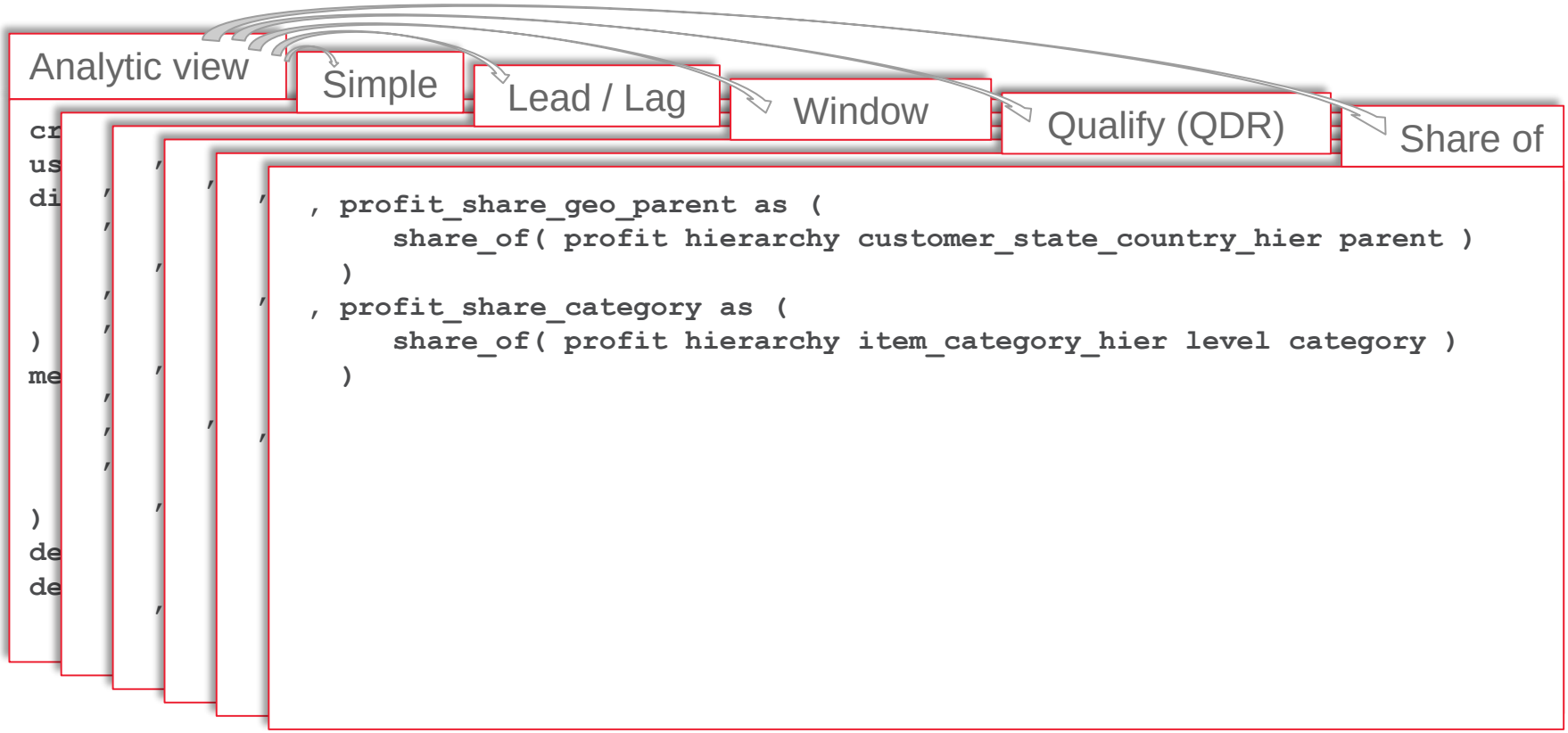
trivadis makes IT easier.

Measure clauses



trivadis makes IT easier.

Measure clauses



trivadis makes IT easier.

Querying with SQL

trivadis
makes IT easier.

Expanded SQL

- Much added analytic syntax within analytic view definitions
- Hierarchies member names allow MDX-like member filter
- SELECT can query from analytic view, specifying desired hierarchies
- Analytic view measures and hierarchy columns can be used as normal column expressions in select list, where clause, etc.
- As it is all metadata, selecting from analytic view and hierarchies is actually rewritten to SQL with joins, aggregations and calculations on the base tables

trivadis makes IT easier.

Query example

Query

```
select date_mth_qtr_yr_hier.member_name as month
     , amount, amount_prior_period, amount_diff_prior_period
     , round(amount_diff_pct_prior_period * 100, 1) as pct
  from sales_av hierarchies (
          date_mth_qtr_yr_hier
       )
 where date_mth_qtr_yr_hier.level_name = 'MONTH'
 order by date_mth_qtr_yr_hier.hier_order;
```


trivadis makes IT easier.

Query example

Query

```
sel
  f
wh
or
```

Output

```
MONTH          AMOUNT AMOUNT_PRIOR_PERIOD AMOUNT_DIFF_PRIOR_PERIOD         PCT
---------- ---------- ------------------- ------------------------ ----------
2014-01     179918.38
2014-02     162298.66           179918.38                -17619.72       -9.8
2014-03      83065.78           162298.66                -79232.88      -48.8
2014-04      79792.79            83065.78                 -3272.99       -3.9
2014-05      82464.93            79792.79                  2672.14        3.3
2014-06     121360.52            82464.93                 38895.59       47.2
2014-07     125733.23           121360.52                  4372.71        3.6
2014-08     126837.02           125733.23                  1103.79         .9
...
2017-09      71217.78           125615.93                -54398.15      -43.3
2017-10      73094.69            71217.78                  1876.91        2.6
2017-11      70147.02            73094.69                 -2947.67         -4
2017-12     179949.06            70147.02                109802.04      156.5

48 rows selected.
```


trivadis makes IT easier.

Query example

Query

```
select date_mth_qtr_yr_hier.member_name as quarter
     , amount, amount_prior_period
     , cost_ytd, cost_ytd_prior_year
  from sales_av hierarchies (
          date_mth_qtr_yr_hier
       )
 where date_mth_qtr_yr_hier.parent_unique_name = '[YEAR].&[2015]'
 order by date_mth_qtr_yr_hier.hier_order;
```


trivadis makes IT easier.

Query example

Query

```
sel
   f
  wh
  or
```

Output

```
QUARTER         AMOUNT AMOUNT_PRIOR_PERIOD   COST_YTD COST_YTD_PRIOR_YEAR
---------- ---------- -------------------- ---------- --------------------
2015-Q1     423988.68            324418.81  148103.48             148394.1
2015-Q2     284102.26            423988.68  244424.98            244437.42
2015-Q3     321613.01            284102.26  352579.29            352676.94
2015-Q4     323018.87            321613.01  466129.48            466399.24
```


trivadis makes IT easier.

Query example

Query

```
select lpad(' ', date_mth_qtr_yr_hier.depth * 2)
       || date_mth_qtr_yr_hier.member_name as period
     , qty, max_qty, cost, cost_qtd
  from sales_av hierarchies (
          date_mth_qtr_yr_hier
       )
 where date_mth_qtr_yr_hier.year_id = 2015
   and date_mth_qtr_yr_hier.is_leaf = 0
 order by date_mth_qtr_yr_hier.hier_order;
```


trivadis makes IT easier.

Query example

Query

```
sel
  f
 wh
 or
```

Output

PERIOD	QTY	MAX_QTY	COST	COST_QTD
2015	25844	14	466129.48	
2015-Q1	7792	14	148103.48	148103.48
2015-01	3469	14	63084.74	63084.74
2015-02	3117	14	56645.04	119729.78
2015-03	1206	7	28373.7	148103.48
2015-Q2	5466	14	96321.5	96321.5
2015-04	1164	7	27785.33	27785.33
2015-05	1206	7	28432.84	56218.17
2015-06	3096	14	40103.33	96321.5
2015-Q3	7291	14	108154.31	108154.31
2015-07	3190	14	41473.46	41473.46
2015-08	3189	14	41514.96	82988.42
2015-09	912	7	25165.89	108154.31
2015-Q4	5295	14	113550.19	113550.19
2015-10	936	7	25682.68	25682.68
2015-11	899	7	24789.31	50471.99
2015-12	3460	14	63078.2	113550.19

trivadis makes IT easier.

Query example

Query

```
select date_mth_qtr_yr_hier.member_name as year
     , qty, amount, amount_next_period, cost, profit
  from sales_av hierarchies (
          date_mth_qtr_yr_hier
       )
 where date_mth_qtr_yr_hier.level_name = 'YEAR'
 order by date_mth_qtr_yr_hier.hier_order;
```


trivadis makes IT easier.

Query example

Query

```
sel
  f
wh
or
```

Output

YEAR	QTY	AMOUNT	AMOUNT_NEXT_PERIOD	COST	PROFIT
2014	25823	1356119.17	1352722.82	466399.24	889719.93
2015	25844	1352722.82	1360671.25	466129.48	886593.34
2016	25968	1360671.25	1357019.68	468413.17	892258.08
2017	25845	1357019.68		467039.17	889980.51

trivadis makes IT easier.

Query example

Query

```
select lpad(' ', item_category_hier.depth * 2)
       || item_category_hier.member_name as item_category
     , qty, amount, profit, margin
  from sales_av hierarchies (
          item_category_hier, date_mth_season_hier
       )
 where date_mth_season_hier.member_unique_name = '[SEASON].&[3]' --Summer
 order by item_category_hier.hier_order;
```


trivadis makes IT easier.

Query example

Query

```
sel
  f
wh
or
```

Output

```
ITEM_CATEGORY              QTY      AMOUNT      PROFIT      MARGIN
-------------- ---------- ---------- ---------- ----------
ALL ITEMS                37876  1494665.2 1001155.45 66.9819201
  For the beach          36402 1457944.92  977999.28 67.0806741
    Bikini                9105  354302.87  264750.11 74.7242352
    Sunglasses            9107  185334.38  120266.14 64.8914357
    Surfboard             9090  774906.81  482142.21 62.2193796
    Volleyball            9100  143400.86  110840.82   77.29439
  Winter sports           1474   36720.28   23156.17 63.0609843
    Bob sled               368    4631.11    3623.35 78.2393422
    Ice skates             367   13153.87    7476.55  56.839166
    Ski gloves             369    2457.34    1121.98 45.6583135
    Snowboard              370   16477.96   10934.29 66.3570612
```


trivadis makes IT easier.

Query example

Query

```
select lpad(' ', customer_state_country_hier.depth * 2)
       || customer_state_country_hier.member_name as cust_state_ctry
     , qty, amount, avg_amount, profit
  from sales_av hierarchies (
          customer_state_country_hier, item_category_hier
       )
 where item_category_hier.member_unique_name = '[CATEGORY].&[2]' --Beach
 order by customer_state_country_hier.hier_order;
```


trivadis makes IT easier.

Query example

Query

```
sel

  f

wh
or
```

Output

```
CUST_STATE_CTRY                QTY     AMOUNT AVG_AMOUNT     PROFIT
---------------------- ---------- ---------- ---------- ----------
ALL CUSTOMERS                52082 2387671.07 179.026098  1594808.4
  Germany                    29797 1163811.78 167.768744  750617.64
    Bayern                   12555  460764.43 172.765066  302102.11
      ABCD GmbH              10013  254860.06 159.387154  158966.79
      Heinz Schubert          2542  205904.37 192.794354  143135.32
    Niedersachsen            17242  703047.35 164.648091  448515.53
      Backhaus Buhl          10021  218548.99 136.507801  122567.05
      Gemeinde Stade          4678  307779.42 192.362138  212103.28
      Heidi Schmidt           2543  176718.94 165.312385   113845.2
  USA                        22285 1223859.29 191.228014  844190.76
    California                9746  807579.34  216.10365  586449.67
      Jack Howell             2538  215165.23 201.465571  152604.75
      Jill Hamilton           2540  215224.69 201.332732  152618.92
      LAPD                    4668  377189.42 235.743388     281226
    New Hampshire            12539  416279.95 156.319921  257741.09
      Sam Flyingbear          2536  186109.84 174.587092  123362.15
      Woods Burgers          10003  230170.11 144.126556  134378.94
```


trivadis makes IT easier.

Query example

Query

```
select lpad(' ', customer_state_country_hier.depth * 2)
       || customer_state_country_hier.member_name as country
     , lpad(' ', item_category_hier.depth * 2)
       || item_category_hier.member_name as category
     , qty, amount
  from sales_av hierarchies (
          customer_state_country_hier
        , item_category_hier
        , date_mth_season_hier
       )
 where customer_state_country_hier.depth <= 1
   and item_category_hier.depth <= 1
   and date_mth_season_hier.member_unique_name = '[MTHOFYR].&[4]' --April
 order by customer_state_country_hier.hier_order
        , item_category_hier.hier_order;
```


trivadis makes IT easier.

Query example

Query

Output

```
sel
  f
  wh
  or
```

```
COUNTRY          CATEGORY                 QTY     AMOUNT
--------------- --------------- ---------- ----------
ALL CUSTOMERS    ALL ITEMS               4668  319763.47
ALL CUSTOMERS      For the beach         3140  210251.43
ALL CUSTOMERS      Winter sports         1528  109512.04
  Germany        ALL ITEMS               2744  162297.46
  Germany          For the beach         1824  103317.01
  Germany          Winter sports          920   58980.45
  USA            ALL ITEMS               1924  157466.01
  USA              For the beach         1316  106934.42
  USA              Winter sports          608   50531.59
```


trivadis makes IT easier.

Performance

trivadis
makes IT easier.

It’s just SQL

- Rewritten to regular SQL means tuning can use anything that can tune SQL

- Indexes
- Index Organized Tables
- In Memory option
- Query Rewrite
  - Pre-aggregation with materialized views

trivadis makes IT easier.

SELECT rewrite

Query

```
select date_mth_qtr_yr_hier.member_name as month
     , amount, amount_prior_period, amount_diff_prior_period
     , round(amount_diff_pct_prior_period * 100, 1) as pct
  from sales_av hierarchies (
         date_mth_qtr_yr_hier
       )
 where date_mth_qtr_yr_hier.level_name = 'MONTH'
 order by date_mth_qtr_yr_hier.hier_order;
```


trivadis makes IT easier.

SELECT rewrite

Query

```
sel
  f
wh
or
```

DBMS_UTILITY.EXPAND_SQL_TEXT output

```
SELECT "A1"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MEMBER_NAME" "MONTH","A1"."MEASURES#AMOUNT" "AMOUNT","A1"."MEASURES#AMOUNT_PRIOR_PER
IOD" "AMOUNT_PRIOR_PERIOD","A1"."MEASURES#AMOUNT_DIFF_PRIOR_PERIOD" "AMOUNT_DIFF_PRIOR_PERIOD",ROUND("A1"."MEASURES#AMOUNT_DIFF_PCT_
PRIOR_PERIOD"*100,1) "PCT" FROM  (SELECT "A33"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MEMBER_NAME" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#
MEMBER_NAME","A33"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#LEVEL_NAME" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#LEVEL_NAME","A33"."DATE_ATTR_
DIM#DATE_MTH_QTR_YR_HIER#HIER_ORDER" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#HIER_ORDER","A34"."MEASURES#AMOUNT" "MEASURES#AMOUNT","A34"
."MEASURES#AMOUNT_PRIOR_PERIOD" "MEASURES#AMOUNT_PRIOR_PERIOD","A34"."MEASURES#AMOUNT_DIFF_PRIOR_PERIOD" "MEASURES#AMOUNT_DIFF_PRIOR
_PERIOD","A34"."MEASURES#AMOUNT_DIFF_PCT_PRIOR_PERIOD" "MEASURES#AMOUNT_DIFF_PCT_PRIOR_PERIOD" FROM  (SELECT "A5"."DATE_ATTR_DIM#DAT
E_MTH_QTR_YR_HIER#DEPTH$" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DEPTH$","A5"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID" "DATE_ATTR_
DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","A5"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID","A5"
."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID","A5"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIE
R#DATE_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DATE_ID","A5"."MEASURES#AMOUNT" "MEASURES#AMOUNT","A4"."MEAS$" "MEASURES#AMOUNT_PRIOR
_PERIOD","A3"."MEAS$" "MEASURES#AMOUNT_DIFF_PRIOR_PERIOD","A2"."MEAS$" "MEASURES#AMOUNT_DIFF_PCT_PRIOR_PERIOD" FROM  (SELECT /*+ NO_
EXPAND */ SUM("SP"."MEASURES#AMOUNT") "MEASURES#AMOUNT","DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#AGGR_TARGETS$"."DATE_ATTR_DIM#DATE_MTH_Q
TR_YR_HIER#DEPTH$" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DEPTH$","DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#AGGR_TARGETS$"."DATE_ATTR_DIM#DAT
E_MTH_QTR_YR_HIER#CYEAR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#AGGR_TARGETS$"."DATE_A
TTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID","DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#AGGR_TARGETS$".
"DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID",CAST(NULL AS date) "DATE_ATTR_DIM#DATE_M
TH_QTR_YR_HIER#DATE_ID" FROM  (SELECT /*+ VECTOR_TRANSFORM */ SUM("A14"."AMOUNT") "MEASURES#AMOUNT","A13"."DATE_ATTR_DIM#DATE_MTH_QT
R_YR_HIER#CYEAR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","A13"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID" "DATE_ATTR_DIM#D
ATE_MTH_QTR_YR_HIER#QTR_ID","A13"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID",CAST(N
ULL AS date) "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DATE_ID" FROM "AV_USER"."SALES_FACT" "A14", (SELECT "A15"."DATE_ATTR_DIM#DATE_ID" "
DATE_ATTR_DIM#DATE_ID","A15"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","A15"."DATE
_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID","A15"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_I
D" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID",CAST(NULL AS date) "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DATE_ID" FROM  (SELECT "A16"
."DATE_ATTR_DIM#DATE_ID" "DATE_ATTR_DIM#DATE_ID","A16"."DATE_ATTR_DIM#CYEAR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","A16".
"DATE_ATTR_DIM#QTR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID","A16"."DATE_ATTR_DIM#MONTH_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIE
R#MONTH_ID" FROM  (SELECT "A32"."DATE_ID" "DATE_ATTR_DIM#DATE_ID","A32"."DATE_NAME" "DATE_ATTR_DIM#DATE_NAME","A32"."MONTH_ID" "DATE
_ATTR_DIM#MONTH_ID","A32"."MONTH_NAME" "DATE_ATTR_DIM#MONTH_NAME","A32"."QTR_ID" "DATE_ATTR_DIM#QTR_ID","A32"."QTR_NAME" "DATE_ATTR_
DIM#QTR_NAME","A32"."CYEAR_ID" "DATE_ATTR_DIM#CYEAR_ID","A32"."CYEAR_NAME" "DATE_ATTR_DIM#CYEAR_NAME","A32"."WEEK_ID" "DATE_ATTR_DIM
#WEEK_ID","A32"."WEEK_NAME" "DATE_ATTR_DIM#WEEK_NAME","A32"."WYEAR_ID" "DATE_ATTR_DIM#WYEAR_ID","A32"."WYEAR_NAME" "DATE_ATTR_DIM#WY
... ... ... ...
... ... ...
... ...
...
```


trivadis makes IT easier.

SELECT rewrite

Query

```
sel
  f
wh
or
```

DBMS_UTILITY.EXPAND_SQL_TEXT output

```
SELECT "A1"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MEMBER_NAME" "MONTH","A1"."MEASURES#AMOUNT" "AMOUNT","A1"."MEASURES#AMOUNT_PRIOR_PER
IOD" "AMOUNT_PRIOR_PERIOD","A1"."MEASURES#AMOUNT_DIFF_PRIOR_PERIOD" "AMOUNT_DIFF_PRIOR_PERIOD",ROUND("A1"."MEASURES#AMOUNT_DIFF_PCT_
PRIOR_PERIOD"*100,1) "PCT" FROM  (SELECT "A33"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MEMBER_NAME" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#
MEMBER_NAME","A33"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#LEVEL_NAME" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#LEVEL_NAME","A33"."DATE_ATTR_
DIM#DATE_MTH_QTR_YR_HIER#HIER_ORDER" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#HIER_ORDER","A34"."MEASURES#AMOUNT" "MEASURES#AMOUNT","A34"
."MEASURES#AMOUNT_PRIOR_PERIOD" "MEASURES#AMOUNT_PRIOR_PERIOD","A34"."MEASURES#AMOUNT_DIFF_PRIOR_PERIOD" "MEASURES#AMOUNT_DIFF_PRIOR
_PERIOD","A34"."MEASURES#AMOUNT_DIFF_PCT_PRIOR_PERIOD" "MEASURES#AMOUNT_DI                                    "DATE_ATTR_DIM#DAT
E_MTH_QTR_YR_HIER#DEPTH$" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DEPTH$","A5"                                    AR_ID" "DATE_ATTR_
DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","A5"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIE                                    R_HIER#QTR_ID","A5"
."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID" "DATE_ATTR_DIM#DATE_MTH_QTR                                    DATE_MTH_QTR_YR_HIE
R#DATE_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DATE_ID","A5"."MEASURES#AMC                                    ASURES#AMOUNT_PRIOR
_PERIOD","A3"."MEAS$" "MEASURES#AMOUNT_DIFF_PRIOR_PERIOD","A2"."MEAS$" "ME                                    M  (SELECT /*+ NO_
EXPAND */ SUM("SP"."MEASURES#AMOUNT") "MEASURES#AMOUNT","DATE_ATTR_DIM#DAT                                    TTR_DIM#DATE_MTH_Q
TR_YR_HIER#DEPTH$" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DEPTH$","DATE_ATTR                                    "DATE_ATTR_DIM#DAT
E_MTH_QTR_YR_HIER#CYEAR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#AGGR_TARGETS$"."DATE_A
TTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID","DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#AGGR_TARGETS$".
"DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID",CAST(NULL AS date) "DATE_ATTR_DIM#DATE_M
TH_QTR_YR_HIER#DATE_ID" FROM  (SELECT /*+ VECTOR_TRANSFORM */ SUM("A14"."AMOUNT") "MEASURES#AMOUNT","A13"."DATE_ATTR_DIM#DATE_MTH_QT
R_YR_HIER#CYEAR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","A13"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID" "DATE_ATTR_DIM#D
ATE_MTH_QTR_YR_HIER#QTR_ID","A13"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID",CAST(N
ULL AS date) "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DATE_ID" FROM "AV_USER"."SALES_FACT" "A14", (SELECT "A15"."DATE_ATTR_DIM#DATE_ID" "
DATE_ATTR_DIM#DATE_ID","A15"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","A15"."DATE
_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID","A15"."DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_I
D" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#MONTH_ID",CAST(NULL AS date) "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#DATE_ID" FROM  (SELECT "A16"
."DATE_ATTR_DIM#DATE_ID" "DATE_ATTR_DIM#DATE_ID","A16"."DATE_ATTR_DIM#CYEAR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#CYEAR_ID","A16".
"DATE_ATTR_DIM#QTR_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIER#QTR_ID","A16"."DATE_ATTR_DIM#MONTH_ID" "DATE_ATTR_DIM#DATE_MTH_QTR_YR_HIE
R#MONTH_ID" FROM  (SELECT "A32"."DATE_ID" "DATE_ATTR_DIM#DATE_ID","A32"."DATE_NAME" "DATE_ATTR_DIM#DATE_NAME","A32"."MONTH_ID" "DATE
_ATTR_DIM#MONTH_ID","A32"."MONTH_NAME" "DATE_ATTR_DIM#MONTH_NAME","A32"."QTR_ID" "DATE_ATTR_DIM#QTR_ID","A32"."QTR_NAME" "DATE_ATTR_
DIM#QTR_NAME","A32"."CYEAR_ID" "DATE_ATTR_DIM#CYEAR_ID","A32"."CYEAR_NAME" "DATE_ATTR_DIM#CYEAR_NAME","A32"."WEEK_ID" "DATE_ATTR_DIM
#WEEK_ID","A32"."WEEK_NAME" "DATE_ATTR_DIM#WEEK_NAME","A32"."WYEAR_ID" "DATE_ATTR_DIM#WYEAR_ID","A32"."WYEAR_NAME" "DATE_ATTR_DIM#WY
... ... ... ...
... ... ...
... ...
...
```

8289108 BYTES of SQL
ABOUT 8 MB

trivadis
makes IT easier.

Materialized views

Query rewrite on materialized views very useful method to gain performance
Syntax supports making analytic view aware of such materialized views

```
create or replace analytic view sales_av
using sales_fact
dimension by (    ((dimension clauses))  )
measures      (    ((measure clauses))    )
default measure       amount
default aggregate by sum
cache    measure group all
         levels (date_mth_qtr_yr_hier.month        ) materialized
         levels (date_wk_yr_hier.week              ) materialized
         levels (date_mth_season_hier.mthofyr      ) materialized
         levels (item_category_hier.category       ) materialized
         levels (customer_group_hier.custgroup     ) materialized
         levels (customer_state_country_hier.state) materialized
;
```

CACHE syntax makes SQL generator of AV select aware of mat.views, so it generates SQL using the MV's (Rather than relying on magic of normal query rewrite)

trivadis makes IT easier.

Classifications

trivadis
makes IT easier.

Classifications

- Metadata of dimensions, hierarchies, analytic views, members, attributes, measures
  - Caption - either in full classification syntax or shortcut
  - Description - either in full classification syntax or shortcut
  - Formatting - only in full classification syntax
- Can be in multiple languages - only in full classification syntax
- **Not** used in querying with SQL
  - But can be queried from dictionary views
- Meant for visualization tools
  - Classifications available via drivers, for example OLE DB for OLAP Provider

trivadis makes IT easier.

Caption and Description shortcuts

Attribute dimension

```
create or replace attribute dimension
item_attr_dim
   caption      'Items'
   description 'Item attribute dimension'
   using item_dim
attributes (
   item_id         caption 'Item id'
 , item_name       caption 'Item name'
 , category_id     caption 'Category id'
 , category_name   caption 'Category name'
)
level item
   caption         'Item'
   description     'Items in product catalogue'
   key             item_id
   alternate key   item_name
   member name     item_name
   member caption item_name
   order by        item_name
   determines      (category_id)
level category
   caption         'Item category'
   description     'Categorization of items'
   key             category_id
   alternate key   category_name
   member name     category_name
   member caption category_name
   order by        category_name
all
   member name     'ALL ITEMS'
   member caption 'All items'
   member description 'All items in catalogue'
```


trivadis makes IT easier.

Caption and Description shortcuts

Attribute dimension

Hierarchy

```
create or replace hierarchy date_mth_qtr_yr_hier
   caption     'D/M/Q/Y Hierarchy'
   description 'Hierarchy day-month-quarter-year'
using date_attr_dim
(
   day
   child of month
   child of quarter
   child of cyear
);
```


trivadis makes IT easier.

Caption and Description shortcuts

Attribute dimension

Hierarchy

Analytic view

```
create or replace analytic view sales_av
   caption     'Sales analysis'
   description 'Analysis of sales by date, item and customer hierarchies'
using sales_fact
dimension by (
   date_attr_dim
      caption     'Sales date dimension'
      description 'Date hierarchies for date of sales'
key          sales_date
...
measures (
   qty          fact  qty
      caption     'Quantity'
      description 'Sold quantity of items'
 , amount       fact  amount
      caption     'Amount'
      description 'Total amount for the sold items'
...
```


trivadis makes IT easier.

Classification long syntax

Basic classifications

```
create or replace analytic view sales_av
   classification caption     value 'Sales analysis'
   classification description value 'Analysis of sales by date, item and customer hierarchies'
using sales_fact
dimension by (
   date_attr_dim
      classification caption      value 'Sales date dimension'
      classification description  value 'Date hierarchies for date of sales'
      classification my_own_class value 'My own dimension classification'
      key         sales_date
      references  date_id
      hierarchies (
...

/*
my_own_class is user-defined classification, may be used by own application
*/
```


trivadis makes IT easier.

Classification long syntax

Basic classifications

```
create or replace analytic view sales_av
...
usi
dim
...
/*
my_
*/
```

Formatting

```
...
measures (
   qty          fact  qty
      classification caption       value 'Quantity'
      classification description   value 'Sold quantity of items'
      classification format_string value '9,999'
      classification fore_color    value '#FF0000'
 , amount       fact  amount
      classification caption       value 'Amount'
      classification description   value 'Total amount for the sold items'
      classification format_string value '$99,999.99'
      classification back_color    value '#00FF00'
...
```


trivadis makes IT easier.

Classification long syntax

Basic classifications

Formatting

Languages

```
create or replace attribute dimension item_attr_dim
   dimension type standard
   classification caption     value 'Items'
   classification caption     value 'Produkte'                    language 'GERMAN'
   classification caption     value 'Varer'                       language 'DANISH'
   classification description value 'Item attribute dimension'
   classification description value 'Produkt Attribut Dimension' language 'GERMAN'
   classification description value 'Vare attribut dimension'    language 'DANISH'
using item_dim
attributes (
...

/*
Not specifying language means "Default" language, which the application may
choose to pick if a classification in the specific name is not available
*/
```


trivadis makes IT easier.

Summary

trivadis
makes IT easier.

Summary

- Analytic views allow dimensional analysis of data
  - On real-time data
  - Without copying data to datawarehouse and cube
  - Without writing complex analytic SQL in individual queries (complexity hidden in metadata allowing analysis queries to be simpler)
- “Standard” SQL tuning methods apply
- Metadata can be enriched with classifications for tool use (OLAP drivers can view the analytic views similar to a cube)

trivadis makes IT easier.

Links



- This presentation PowerPoint http://bit.ly/kibeha_av122_pptx
- Script with all examples from this presentation http://bit.ly/kibeha_av122_sql

trivadis
makes IT easier.

Questions & Answers

Kim Berg Hansen
Senior Consultant

kim.berghansen@trivadis.com

http://bit.ly/kibeha_av122_pptx
http://bit.ly/kibeha_av122_sql



trivadis
makes IT easier.