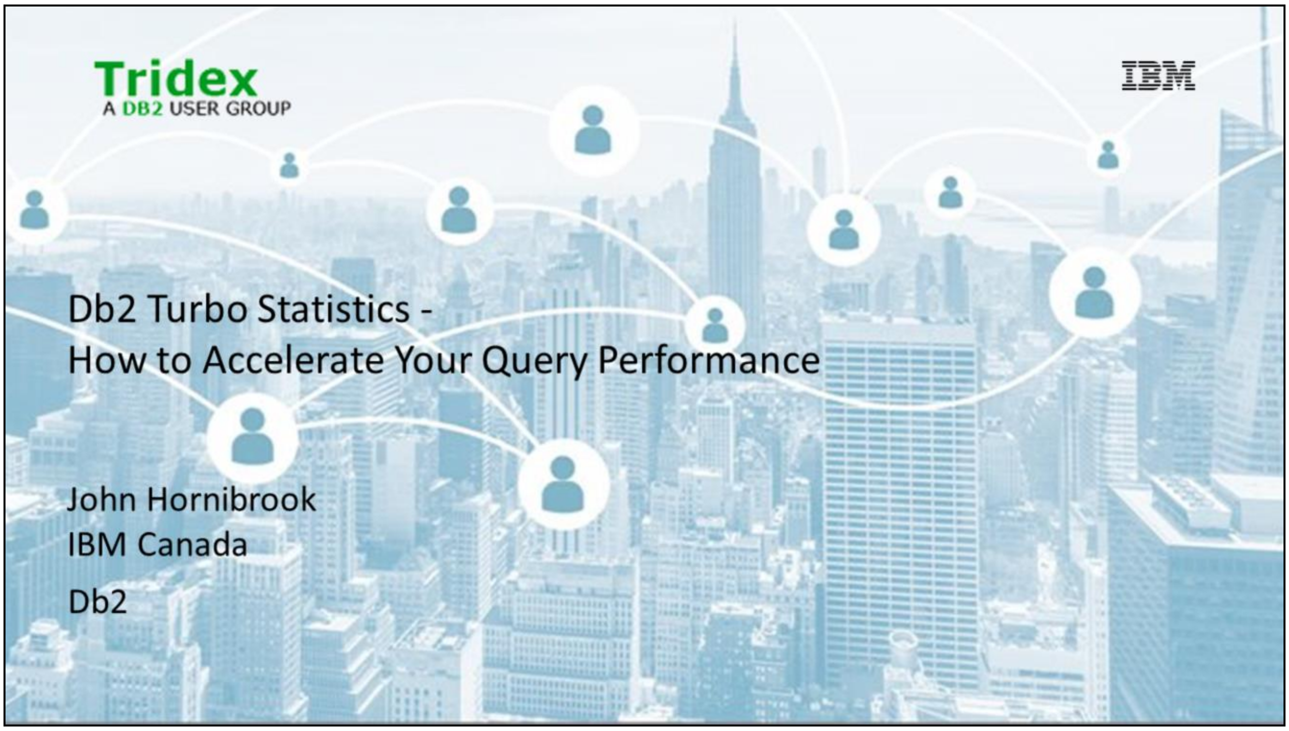
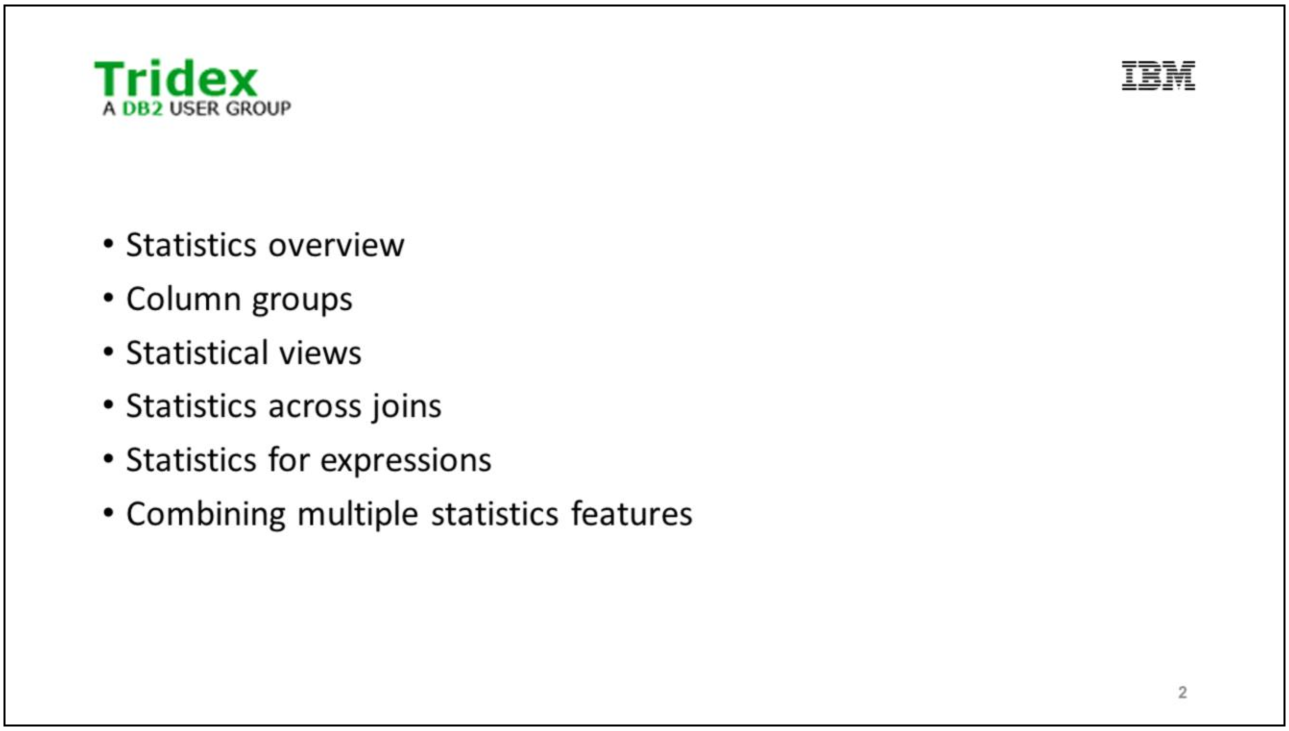


Tridex
A DB2 USER GROUP

IBM

Db2 Turbo Statistics - How to Accelerate Your Query Performance

John Hornibrook
IBM Canada
Db2

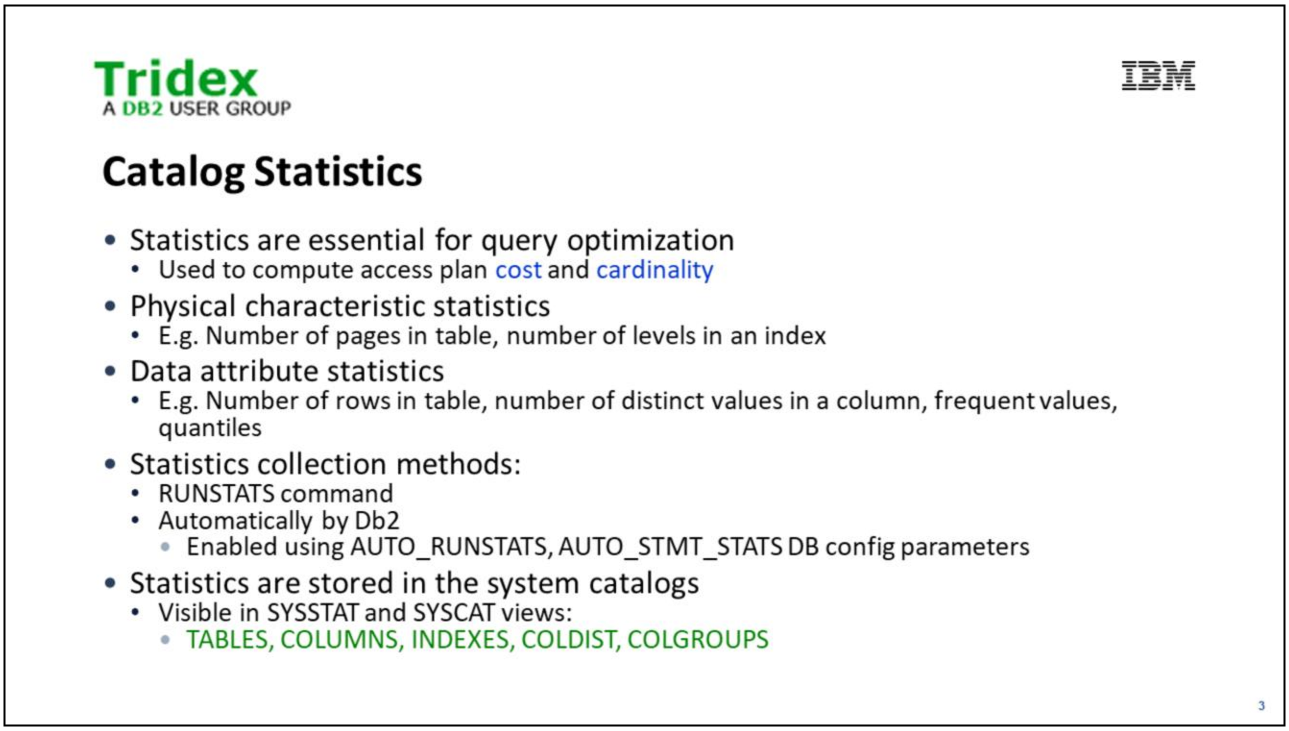


Tridex
A DB2 USER GROUP

IBM

- Statistics overview
- Column groups
- Statistical views
- Statistics across joins
- Statistics for expressions
- Combining multiple statistics features

2


Tridex
A DB2 USER GROUP

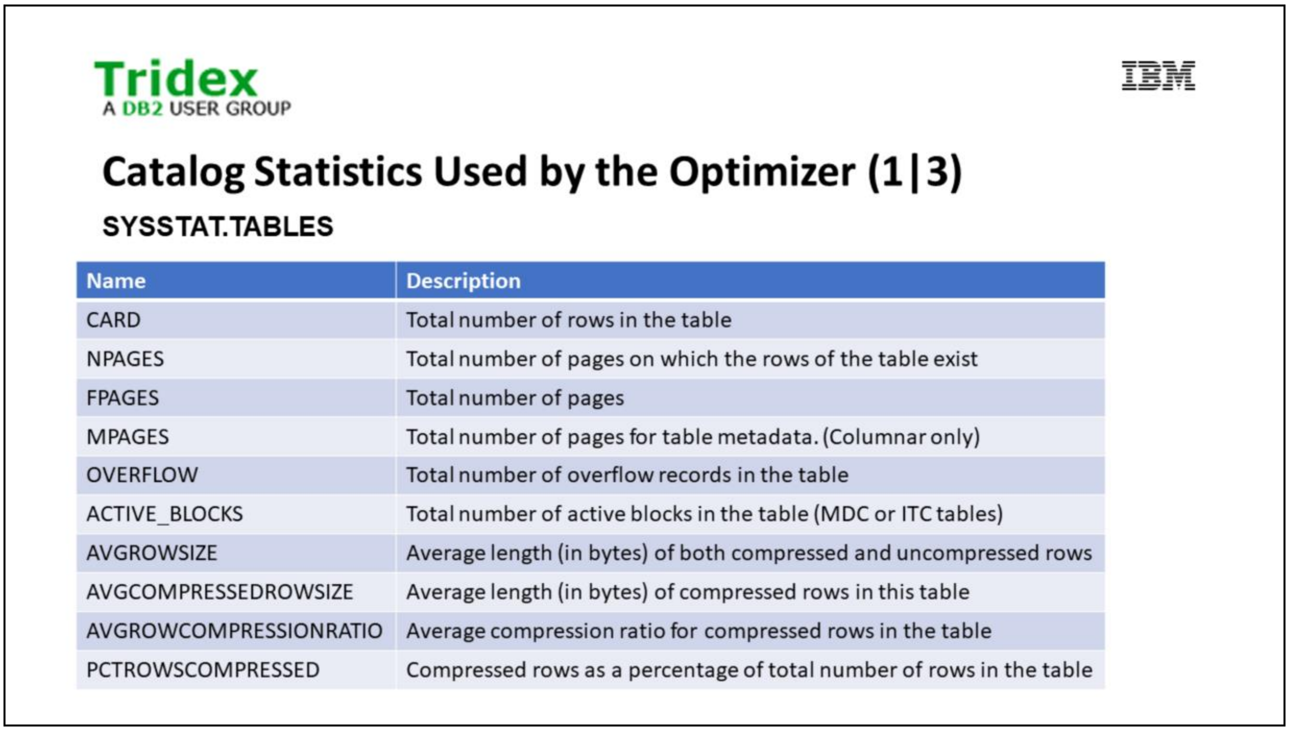
IBM

Catalog Statistics

- Statistics are essential for query optimization
  - Used to compute access plan cost and cardinality
- Physical characteristic statistics
  - E.g. Number of pages in table, number of levels in an index
- Data attribute statistics
  - E.g. Number of rows in table, number of distinct values in a column, frequent values, quantiles
- Statistics collection methods:
  - RUNSTATS command
  - Automatically by Db2
    - Enabled using AUTO_RUNSTATS, AUTO_STMT_STATS DB config parameters
- Statistics are stored in the system catalogs
  - Visible in SYSSTAT and SYSCAT views:
    - TABLES, COLUMNS, INDEXES, COLDIST, COLGROUPS

3

When the SQL compiler optimizes SQL query plans, its decisions are heavily influenced by statistical information about the size of the database tables and indexes. The optimizer also uses information about the distribution of data in specific columns of tables and indexes if these columns are used to select rows or join tables. The optimizer uses this information to estimate the costs of alternative access plans for each query. Db2 will automatically collect statistics depending on how data changes and the statistical needs of queries. Statistics can also be collected manually using the RUNSTATS command. Statistical information is collected for specific tables, indexes and nicknames. The collected statistics are stored in the system catalog tables and can be queried using the SYSSTAT or SYSCAT catalog views.



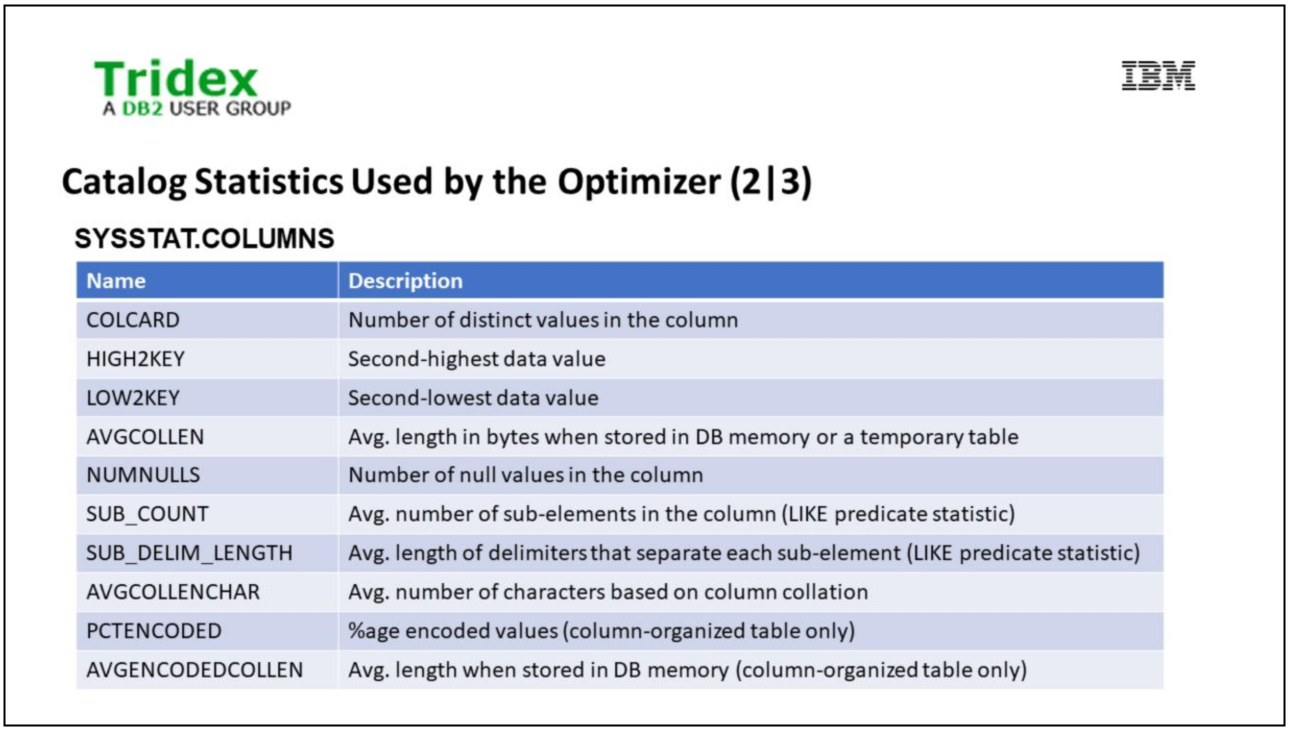
Tridex
A DB2 USER GROUP

IBM

Catalog Statistics Used by the Optimizer (1|3)

SYSSTAT.TABLES

Name	Description
CARD	Total number of rows in the table
NPAGES	Total number of pages on which the rows of the table exist
FPAGES	Total number of pages
MPAGES	Total number of pages for table metadata. (Columnar only)
OVERFLOW	Total number of overflow records in the table
ACTIVE_BLOCKS	Total number of active blocks in the table (MDC or ITC tables)
AVGROWSIZE	Average length (in bytes) of both compressed and uncompressed rows
AVGCOMPRESSEDROWSIZE	Average length (in bytes) of compressed rows in this table
AVGROWCOMPRESSIONRATIO	Average compression ratio for compressed rows in the table
PCTROWSCOMPRESSED	Compressed rows as a percentage of total number of rows in the table



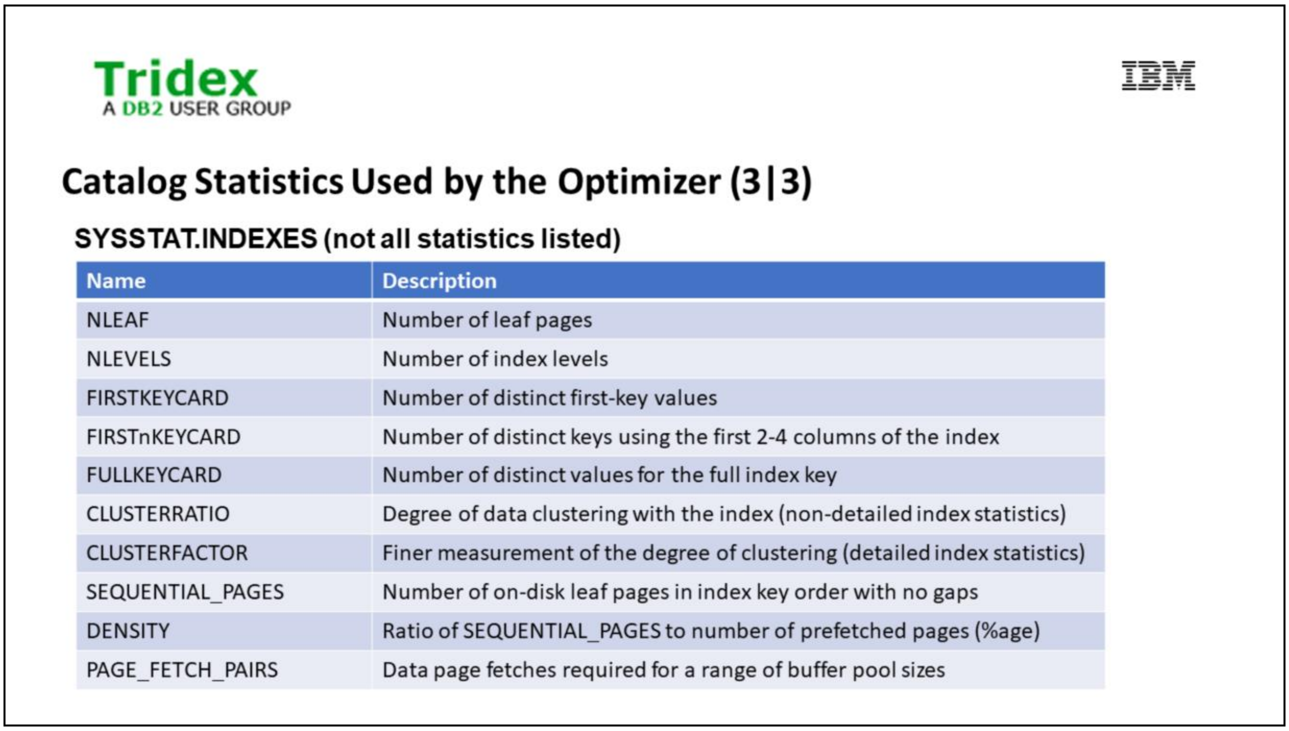
Tridex
A DB2 USER GROUP

IBM

Catalog Statistics Used by the Optimizer (2|3)

SYSSTAT.COLUMNS

Name	Description
COLCARD	Number of distinct values in the column
HIGH2KEY	Second-highest data value
LOW2KEY	Second-lowest data value
AVGCOLLEN	Avg. length in bytes when stored in DB memory or a temporary table
NUMNULLS	Number of null values in the column
SUB_COUNT	Avg. number of sub-elements in the column (LIKE predicate statistic)
SUB_DELIM_LENGTH	Avg. length of delimiters that separate each sub-element (LIKE predicate statistic)
AVGCOLLENCHAR	Avg. number of characters based on column collation
PCTENCODED	%age encoded values (column-organized table only)
AVGENCODEDCOLLEN	Avg. length when stored in DB memory (column-organized table only)



Tridex
A DB2 USER GROUP

Catalog Statistics Used by the Optimizer (3|3)

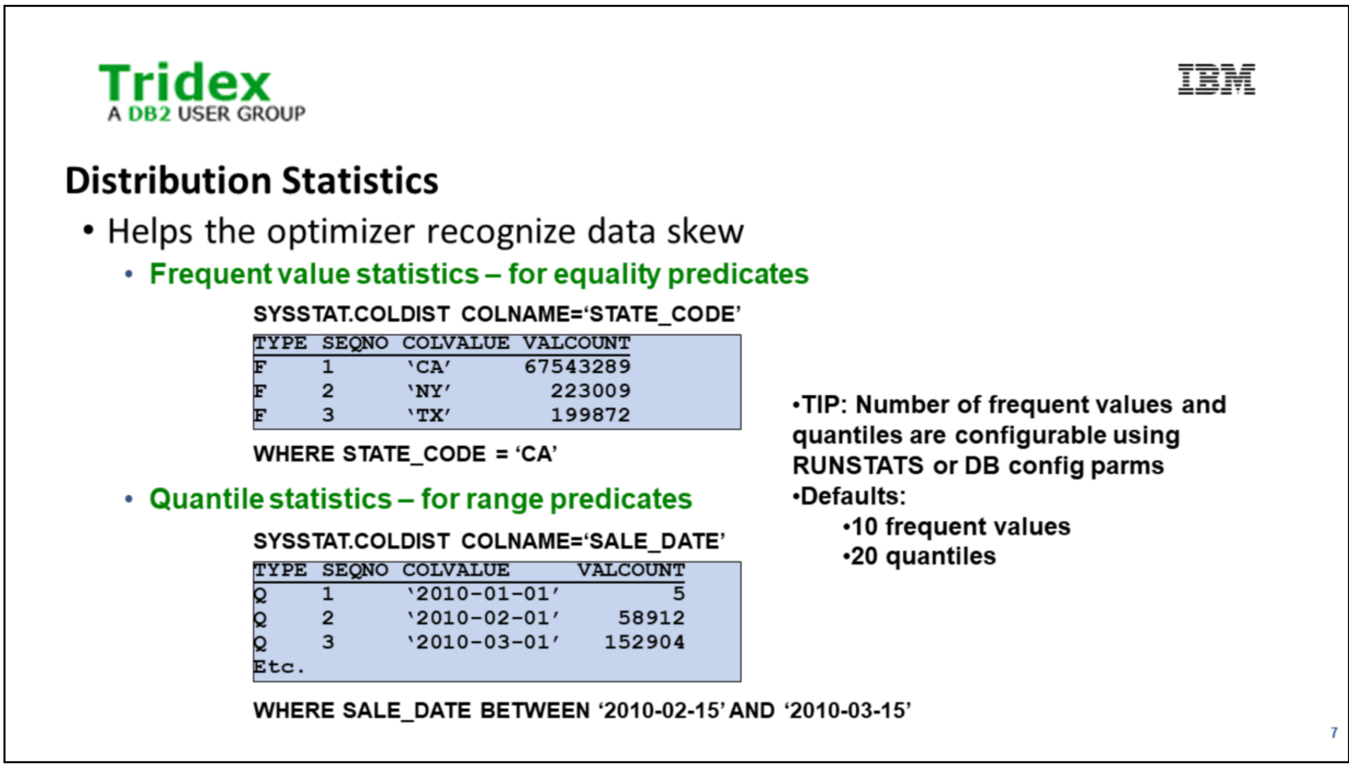
SYSSTAT.INDEXES (not all statistics listed)

Name	Description
NLEAF	Number of leaf pages
NLEVELS	Number of index levels
FIRSTKEYCARD	Number of distinct first-key values
FIRSTnKEYCARD	Number of distinct keys using the first 2-4 columns of the index
FULLKEYCARD	Number of distinct values for the full index key
CLUSTERRATIO	Degree of data clustering with the index (non-detailed index statistics)
CLUSTERFACTOR	Finer measurement of the degree of clustering (detailed index statistics)
SEQUENTIAL_PAGES	Number of on-disk leaf pages in index key order with no gaps
DENSITY	Ratio of SEQUENTIAL_PAGES to number of prefetched pages (%age)
PAGE_FETCH_PAIRS	Data page fetches required for a range of buffer pool sizes

https://www.ibm.com/support/knowledgecenter/en/SSEPGG_11.5.0/com.ibm.db2.luw.sql.ref.doc/doc/r0001072.html

PAGE_FETCH_PAIRS -

A list of pairs of integers, represented in character form. Each pair represents the number of pages in a hypothetical buffer, and the number of page fetches required to scan the table with this index using that hypothetical buffer. Zero-length string if no data is available.



Tridex
A DB2 USER GROUP

IBM

Distribution Statistics

- Helps the optimizer recognize data skew
  - **Frequent value statistics – for equality predicates**

SYSSTAT.COLDIST COLNAME='STATE_CODE'

```
TYPE  SEQNO  COLVALUE  VALCOUNT
F     1       'CA'      67543289
F     2       'NY'        223009
F     3       'TX'        199872
```

WHERE STATE_CODE = 'CA'

  - **Quantile statistics – for range predicates**

SYSSTAT.COLDIST COLNAME='SALE_DATE'

```
TYPE  SEQNO  COLVALUE       VALCOUNT
Q     1       '2010-01-01'         5
Q     2       '2010-02-01'     58912
Q     3       '2010-03-01'    152904
Etc.
```

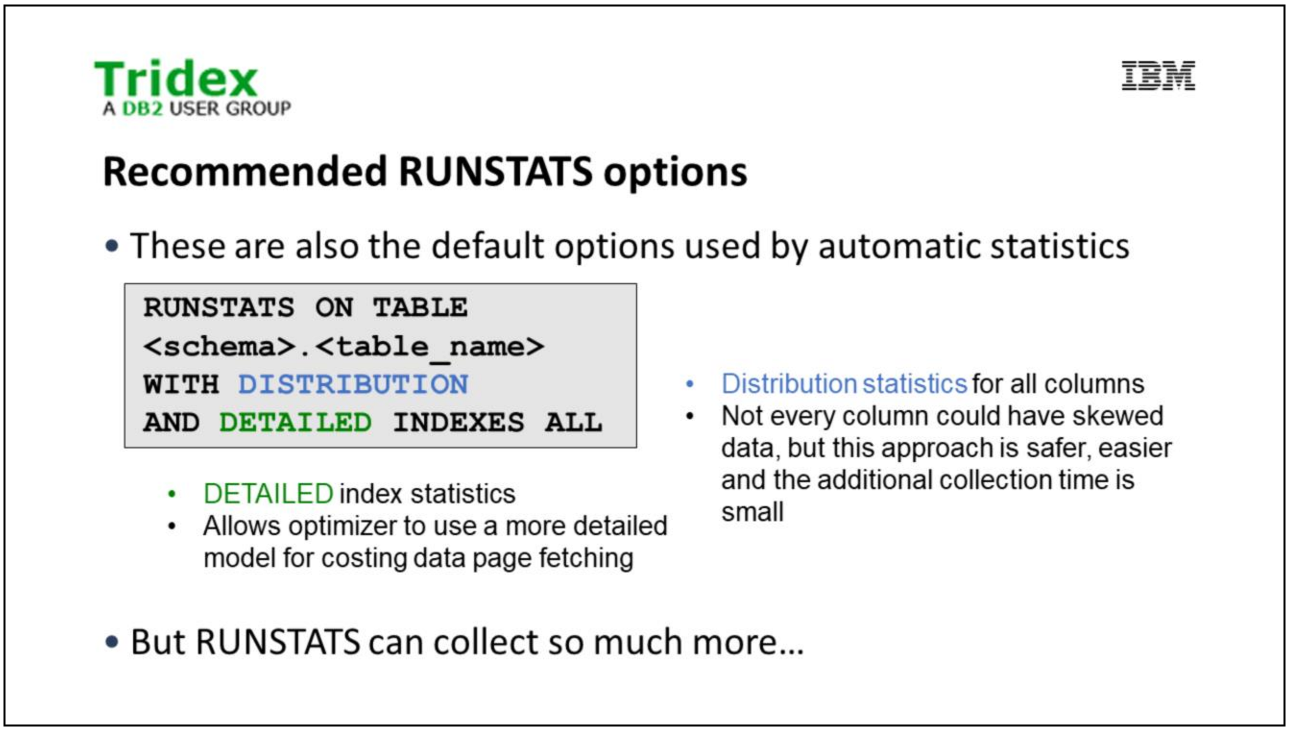
WHERE SALE_DATE BETWEEN '2010-02-15' AND '2010-03-15'

•TIP: Number of frequent values and quantiles are configurable using RUNSTATS or DB config parms

•Defaults:

•10 frequent values

•20 quantiles



Tridex
A DB2 USER GROUP

IBM

Recommended RUNSTATS options

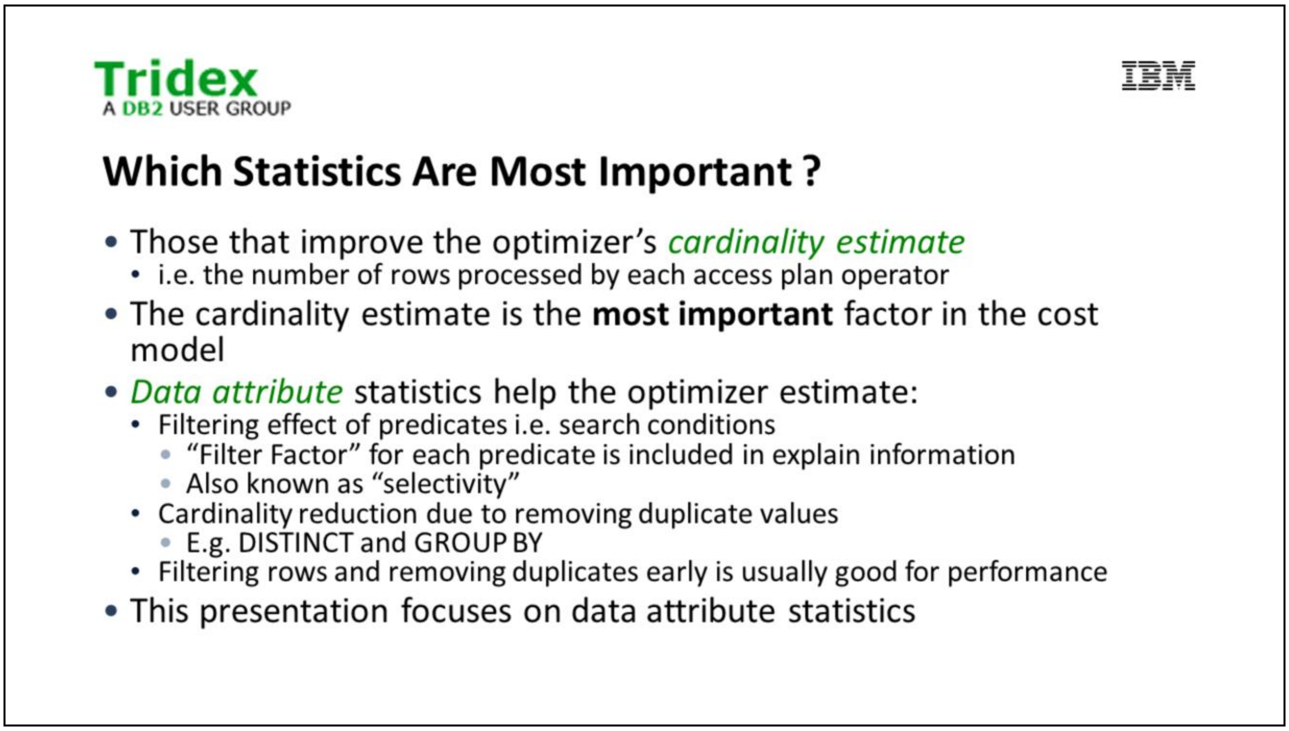
- These are also the default options used by automatic statistics

```
RUNSTATS ON TABLE
<schema>.<table_name>
WITH DISTRIBUTION
AND DETAILED INDEXES ALL
```

- DETAILED index statistics
- Allows optimizer to use a more detailed model for costing data page fetching

- Distribution statistics for all columns
- Not every column could have skewed data, but this approach is safer, easier and the additional collection time is small

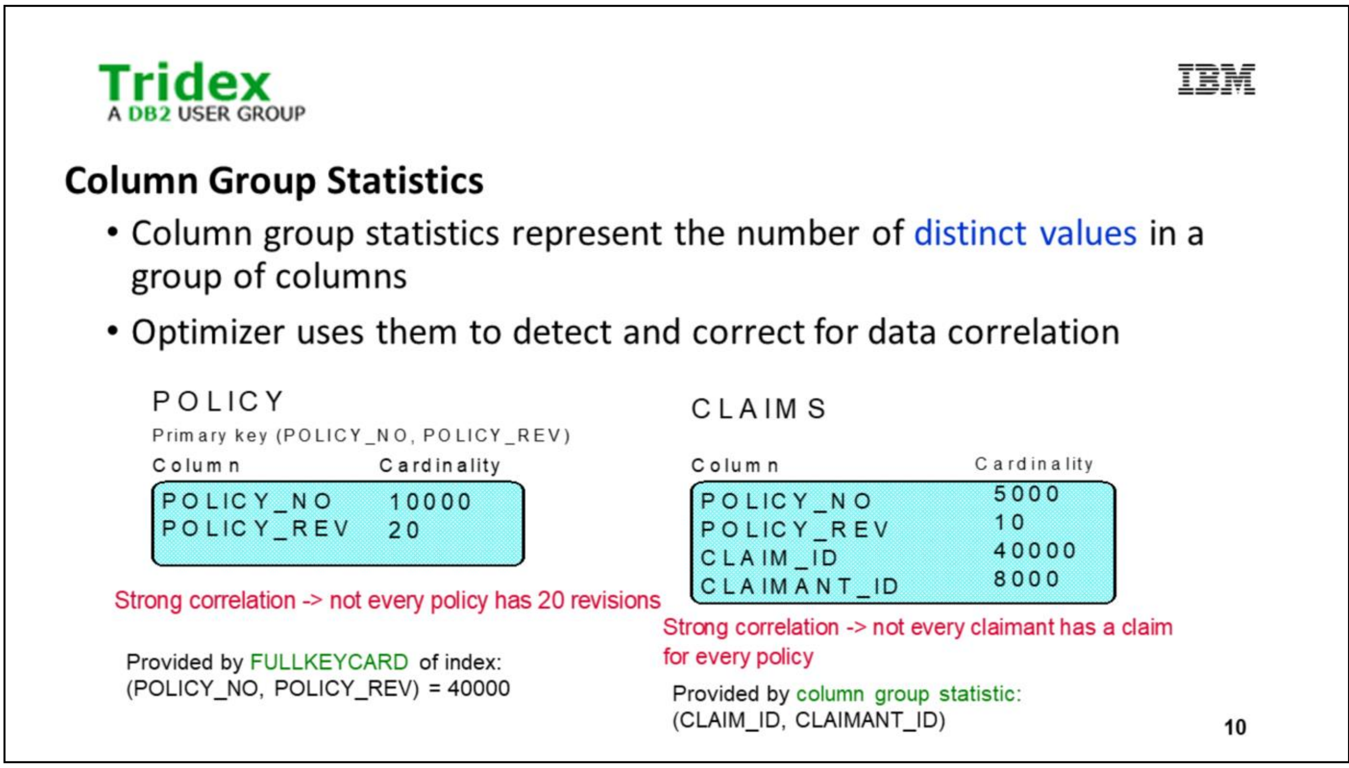
- But RUNSTATS can collect so much more…


Tridex
A DB2 USER GROUP

IBM

Which Statistics Are Most Important ?

- Those that improve the optimizer's *cardinality estimate*
  - i.e. the number of rows processed by each access plan operator
- The cardinality estimate is the **most important** factor in the cost model
- *Data attribute* statistics help the optimizer estimate:
  - Filtering effect of predicates i.e. search conditions
    - "Filter Factor" for each predicate is included in explain information
    - Also known as "selectivity"
  - Cardinality reduction due to removing duplicate values
    - E.g. DISTINCT and GROUP BY
  - Filtering rows and removing duplicates early is usually good for performance
- This presentation focuses on data attribute statistics



Tridex
A DB2 USER GROUP

IBM

Column Group Statistics

- Column group statistics represent the number of distinct values in a group of columns
- Optimizer uses them to detect and correct for data correlation

POLICY

Primary key (POLICY_NO, POLICY_REV)

Column	Cardinality
POLICY_NO	10000
POLICY_REV	20

Strong correlation -> not every policy has 20 revisions

Provided by FULLKEYCARD of index:
(POLICY_NO, POLICY_REV) = 40000

CLAIMS

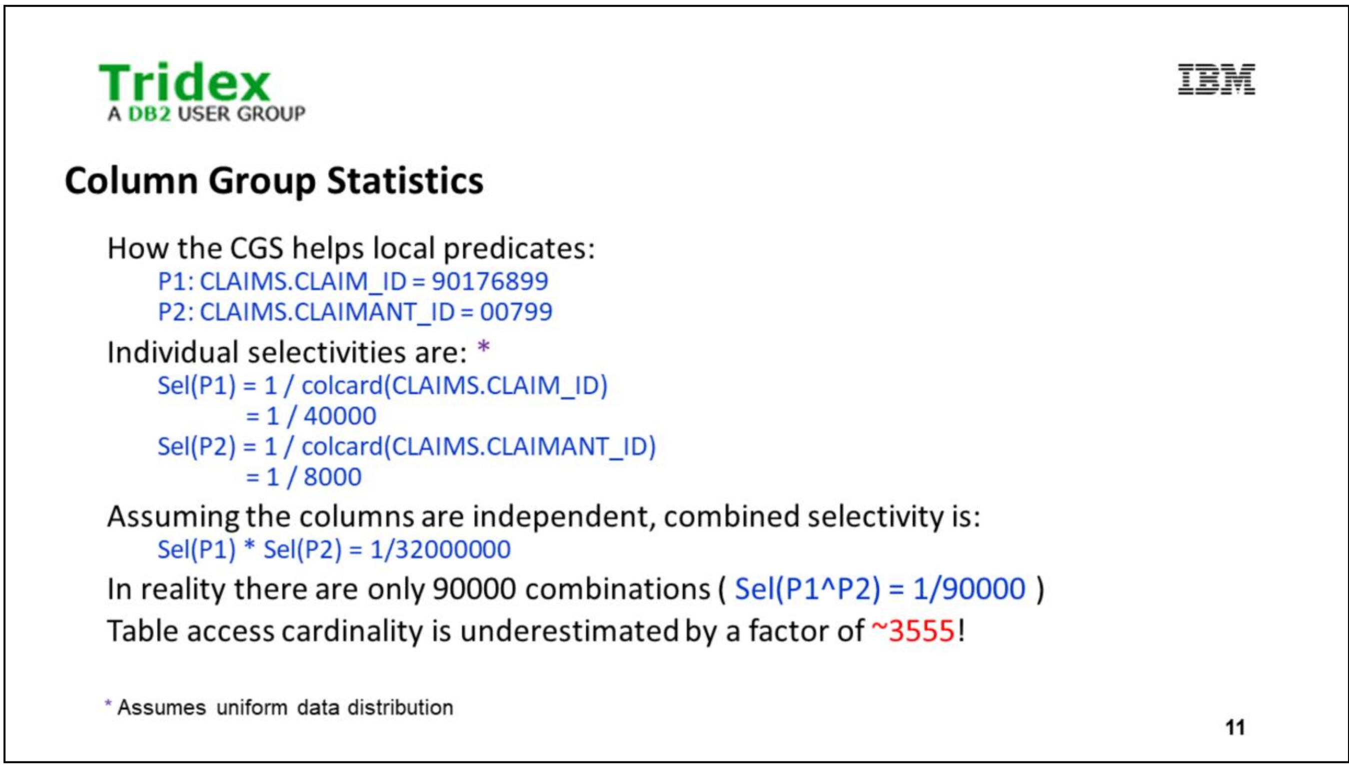
Column	Cardinality
POLICY_NO	5000
POLICY_REV	10
CLAIM_ID	40000
CLAIMANT_ID	8000

Strong correlation -> not every claimant has a claim for every policy

Provided by column group statistic:
(CLAIM_ID, CLAIMANT_ID)

10

Column group cardinality statistics represent the number of distinct values in a group of columns. This statistic helps the optimizer recognize when the data in columns is correlated. Without it, the optimizer assumes that the data in columns is independent.



Tridex
A DB2 USER GROUP

IBM

Column Group Statistics

How the CGS helps local predicates:

P1: CLAIMS.CLAIM_ID = 90176899
P2: CLAIMS.CLAIMANT_ID = 00799

Individual selectivities are: *

Sel(P1) = 1 / colcard(CLAIMS.CLAIM_ID)
= 1 / 40000
Sel(P2) = 1 / colcard(CLAIMS.CLAIMANT_ID)
= 1 / 8000

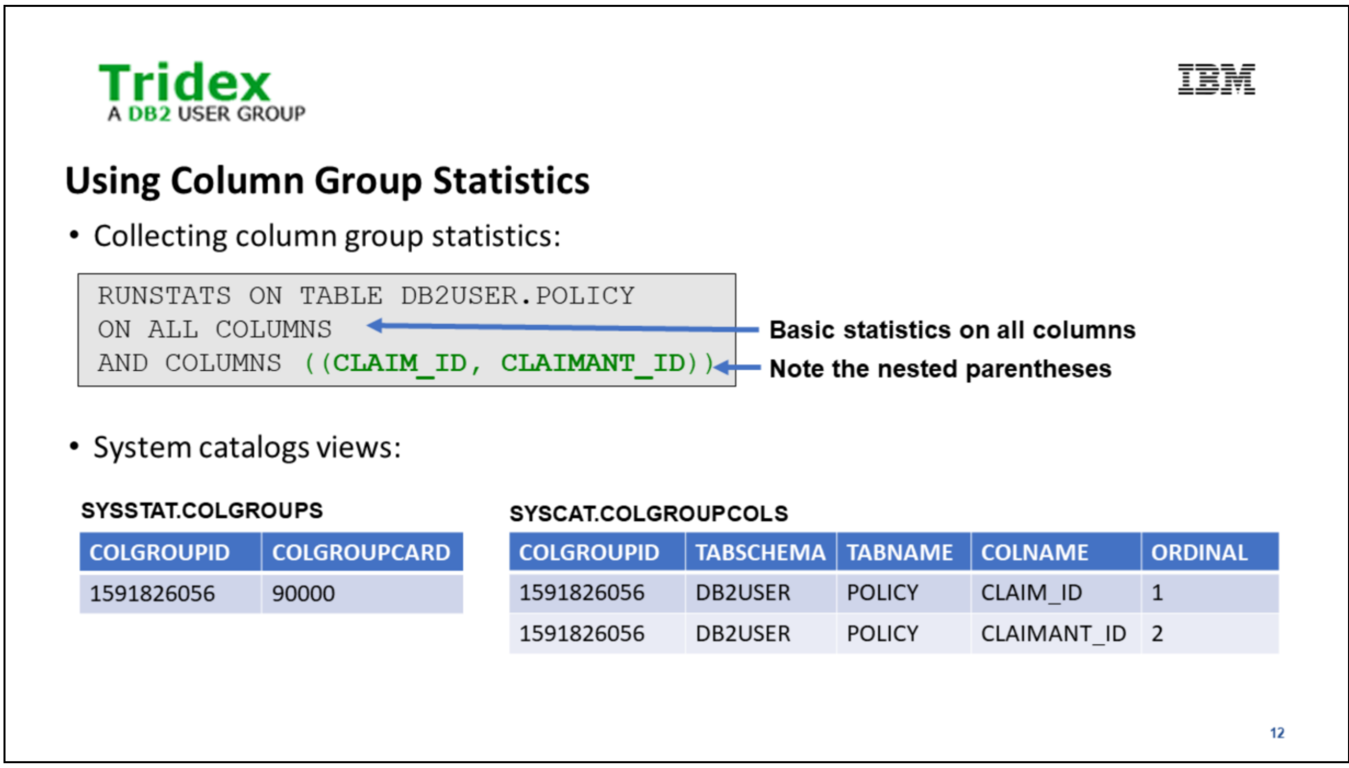
Assuming the columns are independent, combined selectivity is:

Sel(P1) * Sel(P2) = 1/32000000

In reality there are only 90000 combinations (Sel(P1^P2) = 1/90000)

Table access cardinality is underestimated by a factor of ~3555!

* Assumes uniform data distribution


Tridex
A DB2 USER GROUP

IBM

Using Column Group Statistics

- Collecting column group statistics:

```
RUNSTATS ON TABLE DB2USER.POLICY
ON ALL COLUMNS
AND COLUMNS ((CLAIM_ID, CLAIMANT_ID))
```

Basic statistics on all columns
Note the nested parentheses

- System catalogs views:

SYSSTAT.COLGROUPS

COLGROUPID	COLGROUPCARD
1591826056	90000

SYSCAT.COLGROUPCOLS

COLGROUPID	TABSCHEMA	TABNAME	COLNAME	ORDINAL
1591826056	DB2USER	POLICY	CLAIM_ID	1
1591826056	DB2USER	POLICY	CLAIMANT_ID	2

12

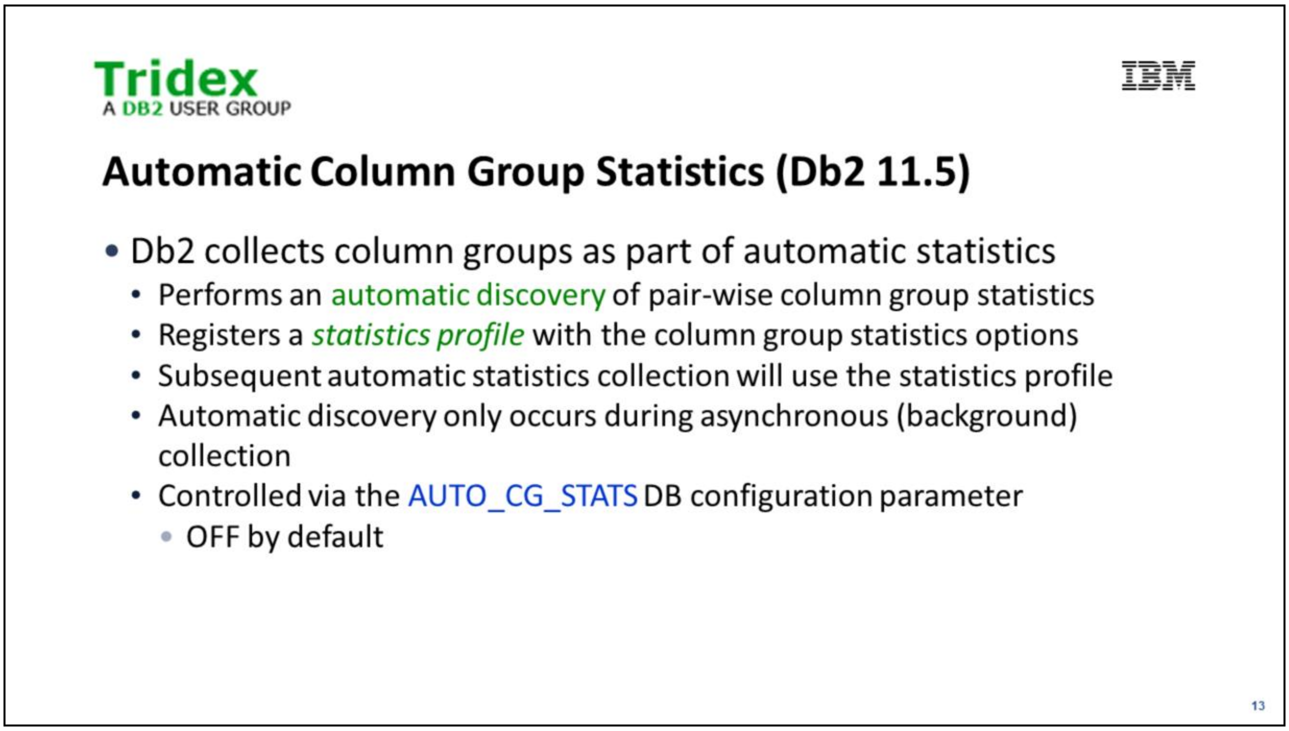
-- Example query to find all the column groups for a particular table

select colgroupschema, colgroupname, g.colgroupid, colname, ordinal, colgroupcard

from sysstat.colgroups g, syscat.colgroupcols c

where g.colgroupid = c.colgroupid and c.tabname = ‘<tabname>’

order by 3,5


Tridex
A DB2 USER GROUP

IBM

Automatic Column Group Statistics (Db2 11.5)

- Db2 collects column groups as part of automatic statistics
  - Performs an automatic discovery of pair-wise column group statistics
  - Registers a *statistics profile* with the column group statistics options
  - Subsequent automatic statistics collection will use the statistics profile
  - Automatic discovery only occurs during asynchronous (background) collection
  - Controlled via the AUTO_CG_STATS DB configuration parameter
    - OFF by default

13

https://www.ibm.com/support/knowledgecenter/en/SSEPGG_11.5.0/com.ibm.db2.luw.admin.perf.doc/doc/c0055085.html

The optimizer uses column group statistics to account for statistical correlation when estimating the combined selectivity of multiple predicates and when computing the number of distinct groupings for operations that group data such as GROUP BY or DISTINCT. Gathering column group statistics can be automated through the automatic statistics collection feature in Db2. Enabling or disabling the automatic collection of column group statistics is done by using the auto_cg_stats database configuration parameter. To enable this function, issue the following command: update db cfg for *dbname* using auto_cg_stats on

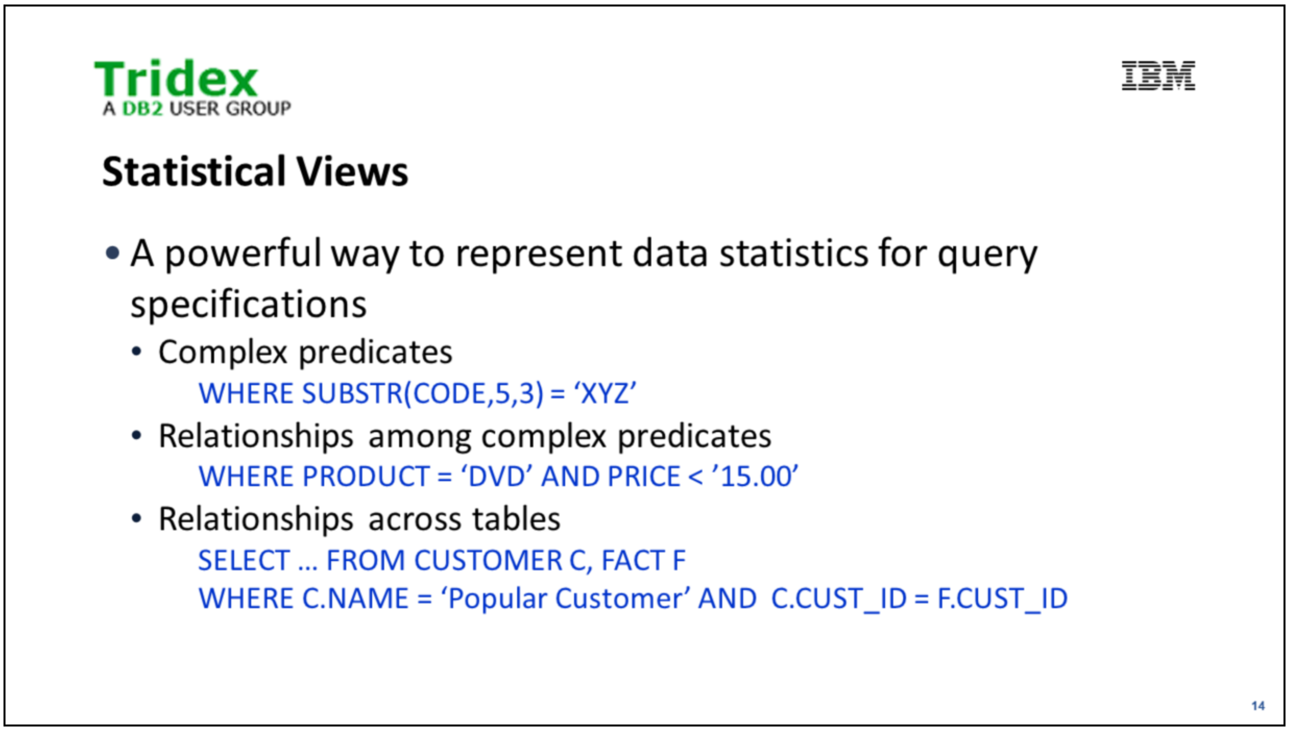
The automatic collection of column group statistics will generate a profile describing the statistics that need to be collected. If a user profile does not exist, the background statistics collection will initially perform an automatic discovery of pair-wise column group statistics within the table and set a statistics profile. After the discovery is completed, statistics are gathered on the table using the existing statistics profile feature. The set of column groups discovered is preserved across subsequent discoveries.

If a statistics profile is already manually set, it will be used as is and the discovery is not performed. The automatically generated statistics profile can be used together with any PROFILE option of the RUNSTATS command. If the profile is updated using the UPDATE PROFILE option, any further discovery is blocked on the table, but the set of column group statistics already set in the profile will continue to be collected automatically as well as with a manual RUNSTATS that includes the USE PROFILE option.

The UNSET PROFILE command can be used to remove the statistics profile to restart the discovery process.

To disable this feature, issue the following command: update db cfg for *dbname* using auto_cg_stats off

Disabling this feature will prevent any further discovery, but the statistic profiles will persist and will continue to be used. If there is a need to remove the profile, use the UNSET PROFILE option of RUNSTATS.



Tridex
A DB2 USER GROUP

IBM

Statistical Views

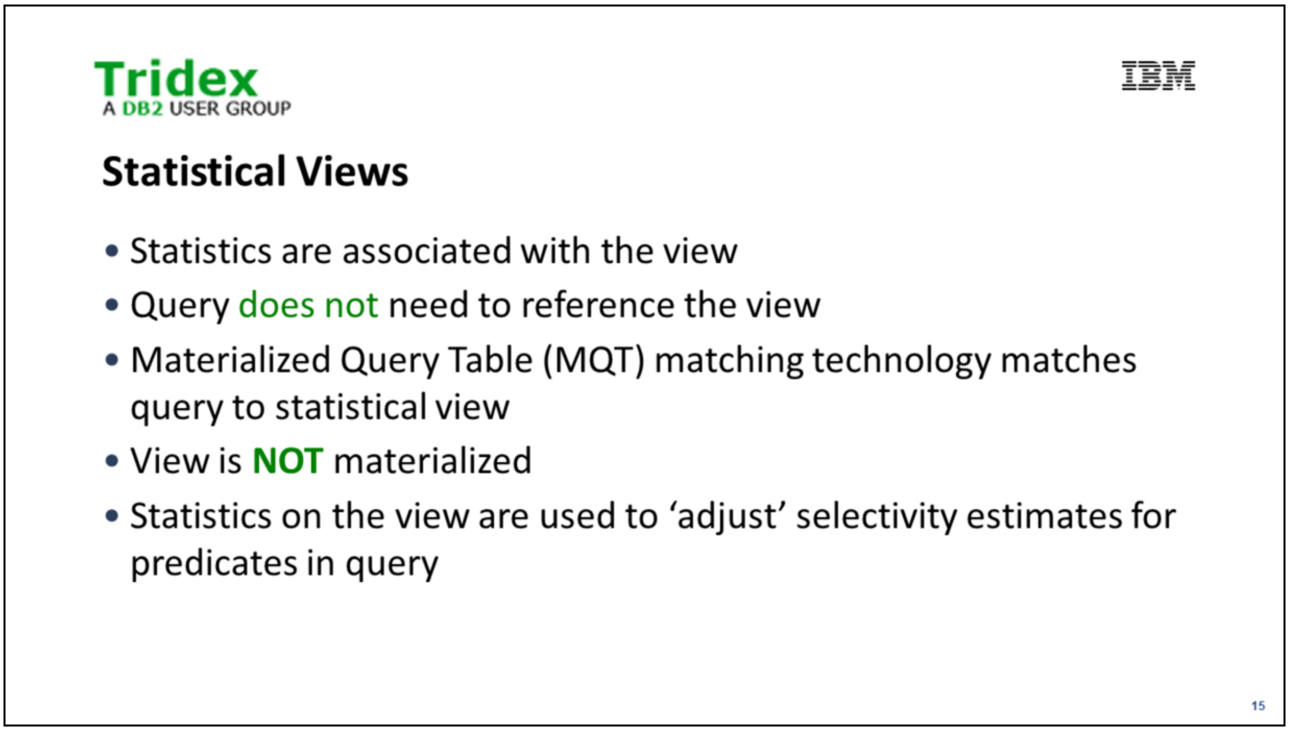
- A powerful way to represent data statistics for query specifications
  - Complex predicates

    ```
    WHERE SUBSTR(CODE,5,3) = 'XYZ'
    ```
  - Relationships among complex predicates

    ```
    WHERE PRODUCT = 'DVD' AND PRICE < '15.00'
    ```
  - Relationships across tables

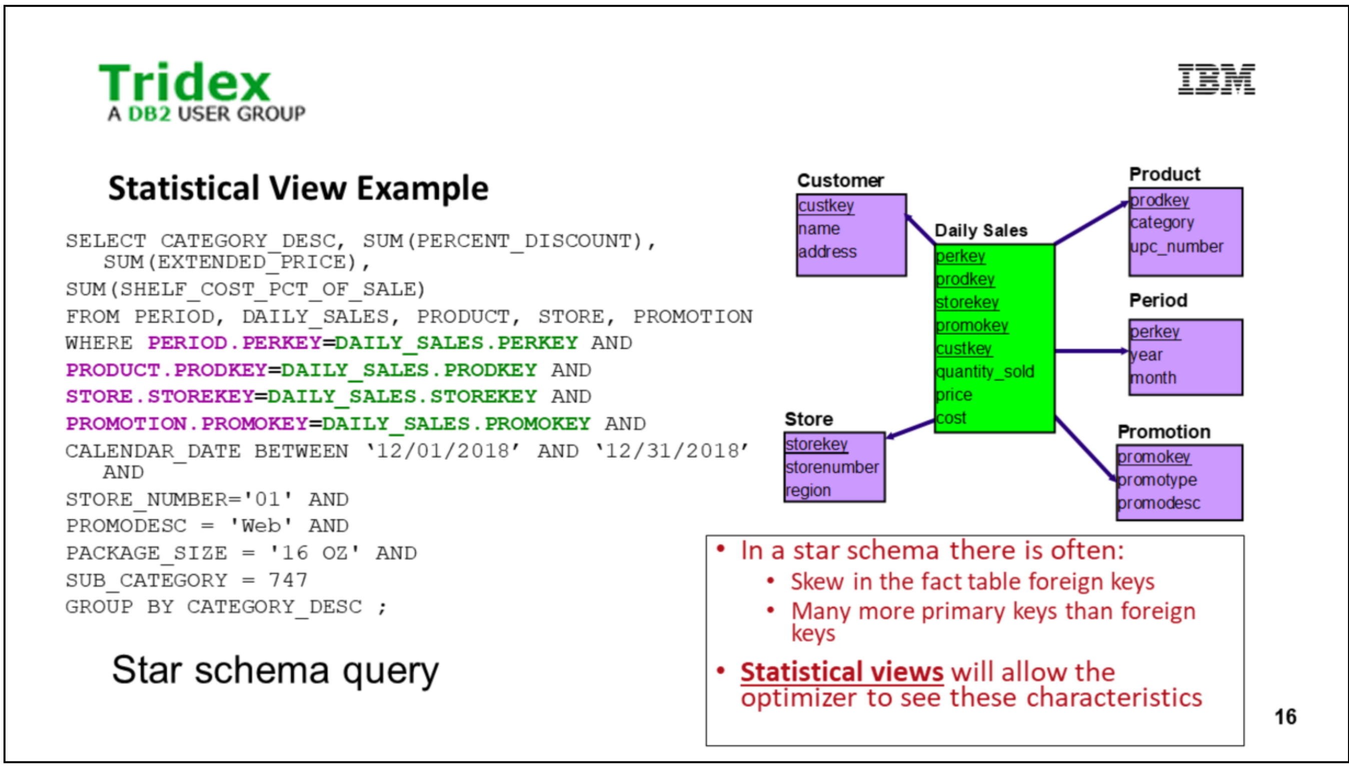
    ```
    SELECT … FROM CUSTOMER C, FACT F
    WHERE C.NAME = 'Popular Customer' AND  C.CUST_ID = F.CUST_ID
    ```

14



Statistical Views

- Statistics are associated with the view
- Query does not need to reference the view
- Materialized Query Table (MQT) matching technology matches query to statistical view
- View is **NOT** materialized
- Statistics on the view are used to ‘adjust’ selectivity estimates for predicates in query



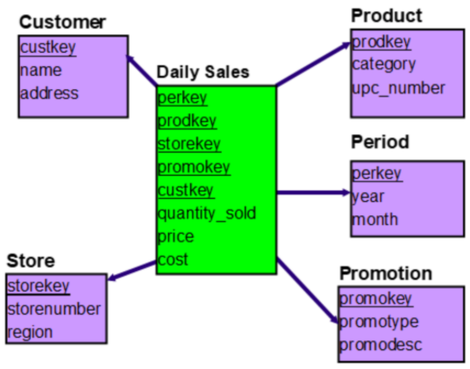
Tridex
A DB2 USER GROUP

IBM

Statistical View Example

```
SELECT CATEGORY_DESC, SUM(PERCENT_DISCOUNT),
   SUM(EXTENDED_PRICE),
SUM(SHELF_COST_PCT_OF_SALE)
FROM PERIOD, DAILY_SALES, PRODUCT, STORE, PROMOTION
WHERE PERIOD.PERKEY=DAILY_SALES.PERKEY AND
PRODUCT.PRODKEY=DAILY_SALES.PRODKEY AND
STORE.STOREKEY=DAILY_SALES.STOREKEY AND
PROMOTION.PROMOKEY=DAILY_SALES.PROMOKEY AND
CALENDAR_DATE BETWEEN '12/01/2018' AND '12/31/2018'
   AND
STORE_NUMBER='01' AND
PROMODESC = 'Web' AND
PACKAGE_SIZE = '16 OZ' AND
SUB_CATEGORY = 747
GROUP BY CATEGORY_DESC ;
```

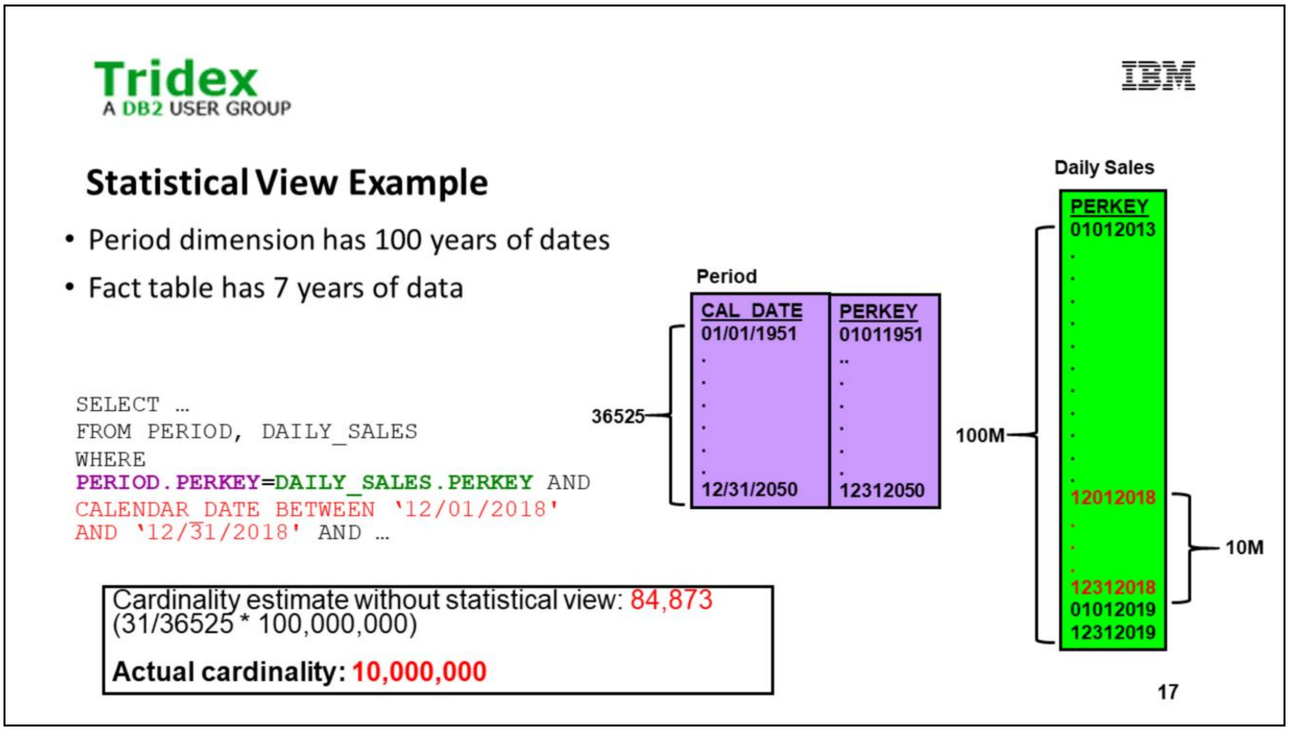
Star schema query



- In a star schema there is often:
  - Skew in the fact table foreign keys
  - Many more primary keys than foreign keys
- **Statistical views** will allow the optimizer to see these characteristics

16

Start by understanding the underlying schema used by the query. In this example, it is a star schema, which has characteristics that can sometimes pose query optimization challenges. You can use statistical views to capture more complex relationships across the fact and dimension tables. This makes the optimizer aware of skew in the fact table foreign key columns and that the fact table foreign keys only contain a subset of the dimension primary key values.



Tridex
A DB2 USER GROUP

IBM

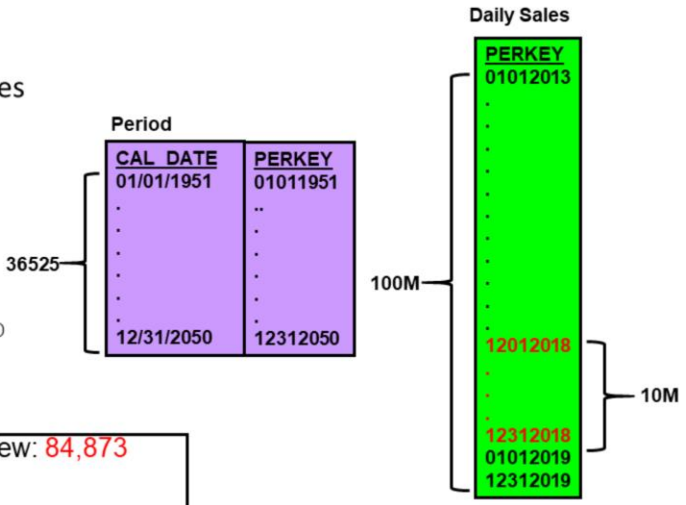
Statistical View Example

- Period dimension has 100 years of dates
- Fact table has 7 years of data

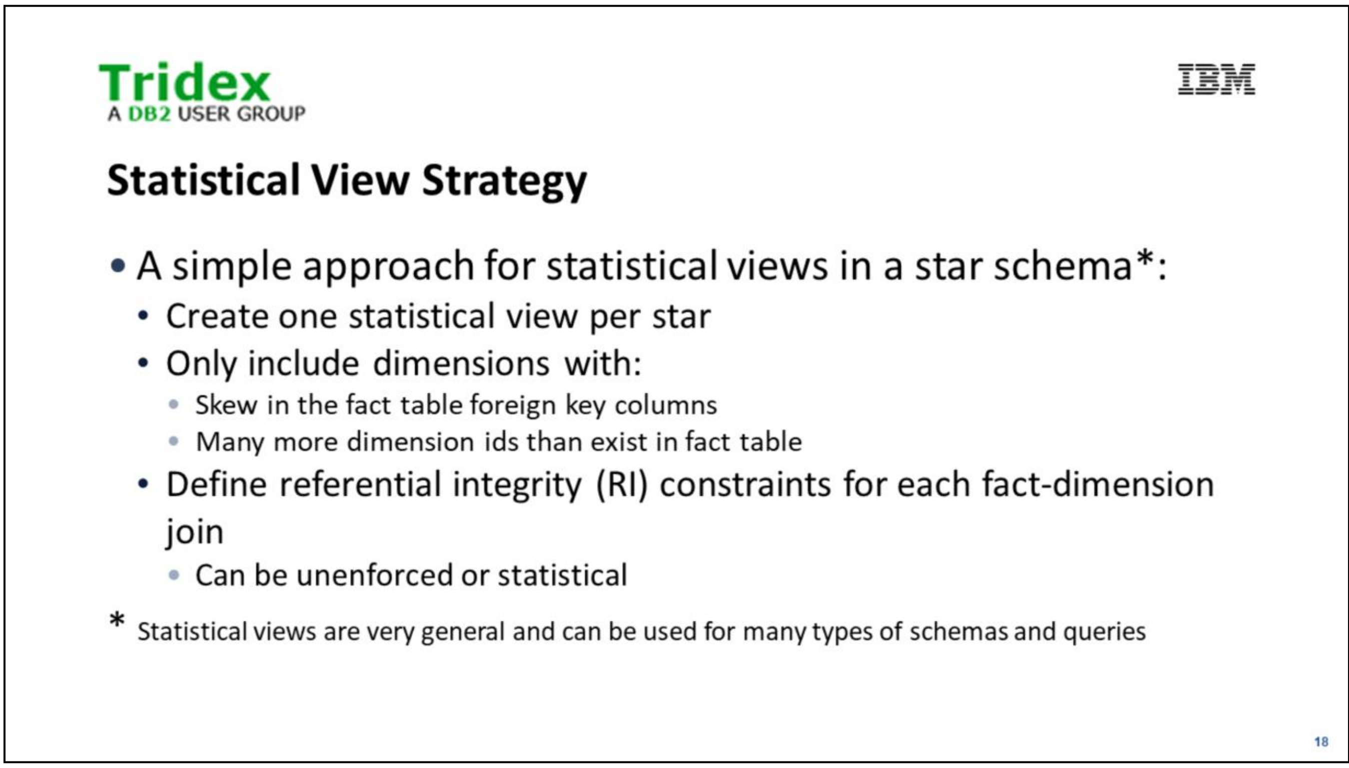
```
SELECT …
FROM PERIOD, DAILY_SALES
WHERE
PERIOD.PERKEY=DAILY_SALES.PERKEY AND
CALENDAR_DATE BETWEEN '12/01/2018'
AND '12/31/2018' AND …
```

Cardinality estimate without statistical view: 84,873
(31/36525 * 100,000,000)

Actual cardinality: 10,000,000



17



Tridex
A DB2 USER GROUP

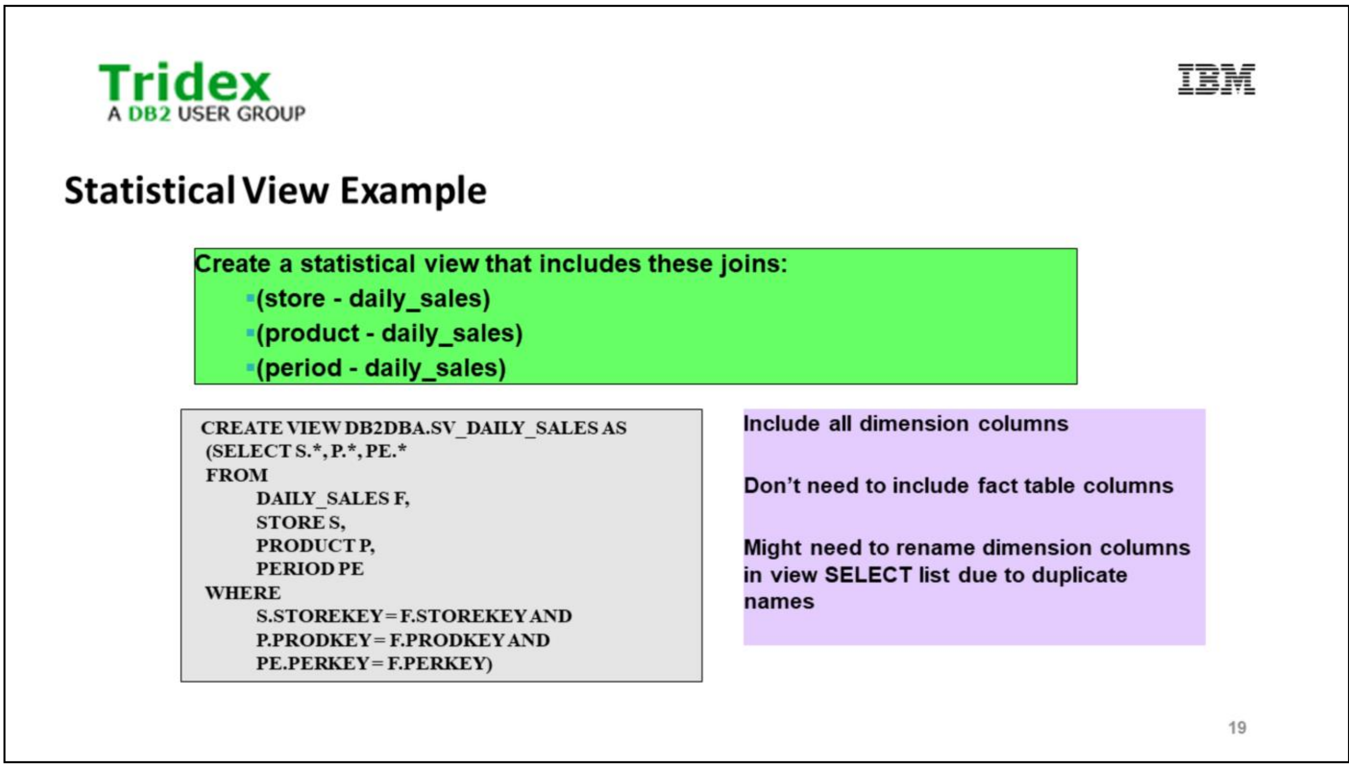
IBM

Statistical View Strategy

- A simple approach for statistical views in a star schema*:
  - Create one statistical view per star
  - Only include dimensions with:
    - Skew in the fact table foreign key columns
    - Many more dimension ids than exist in fact table
  - Define referential integrity (RI) constraints for each fact-dimension join
    - Can be unenforced or statistical

* Statistical views are very general and can be used for many types of schemas and queries

18



Statistical View Example

Create a statistical view that includes these joins:

- (store - daily_sales)
- (product - daily_sales)
- (period - daily_sales)

```
CREATE VIEW DB2DBA.SV_DAILY_SALES AS
 (SELECT S.*, P.*, PE.*
 FROM
        DAILY_SALES F,
        STORE S,
        PRODUCT P,
        PERIOD PE
 WHERE
        S.STOREKEY = F.STOREKEY AND
        P.PRODKEY = F.PRODKEY AND
        PE.PERKEY = F.PERKEY)
```

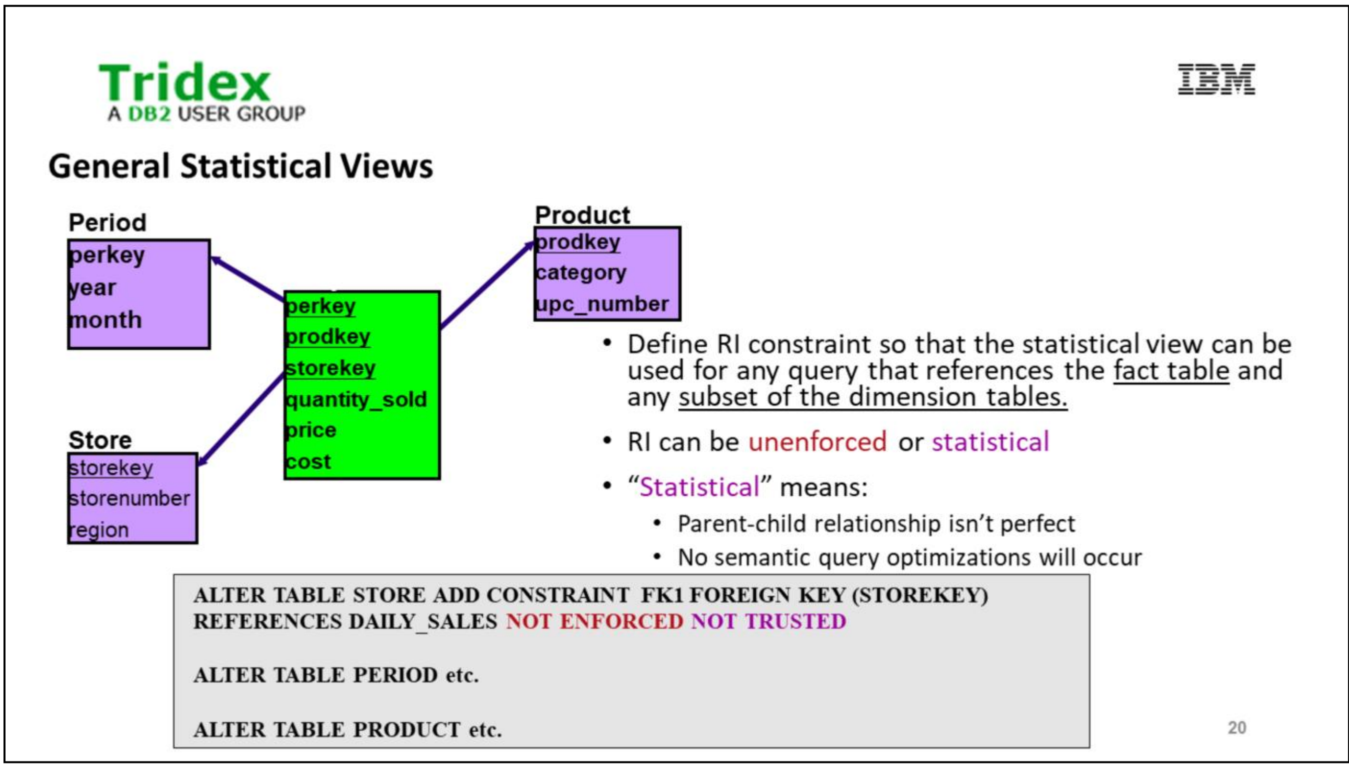
Include all dimension columns

Don't need to include fact table columns

Might need to rename dimension columns in view **SELECT** list due to duplicate names

19

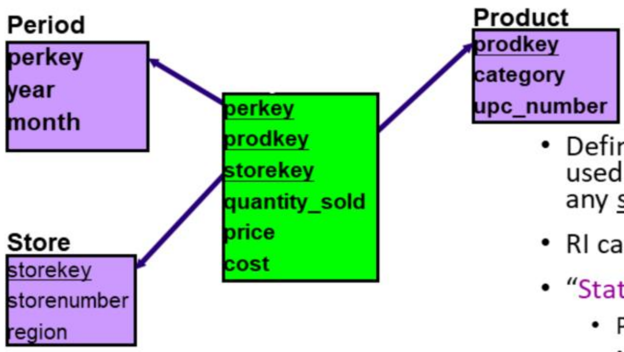
A statistical view is created by first creating a view and then enabling it for optimization using the ALTER VIEW statement. RUNSTATS is then run on the statistical view, populating the system catalog tables with statistics for the view.



Tridex
A DB2 USER GROUP

IBM

General Statistical Views



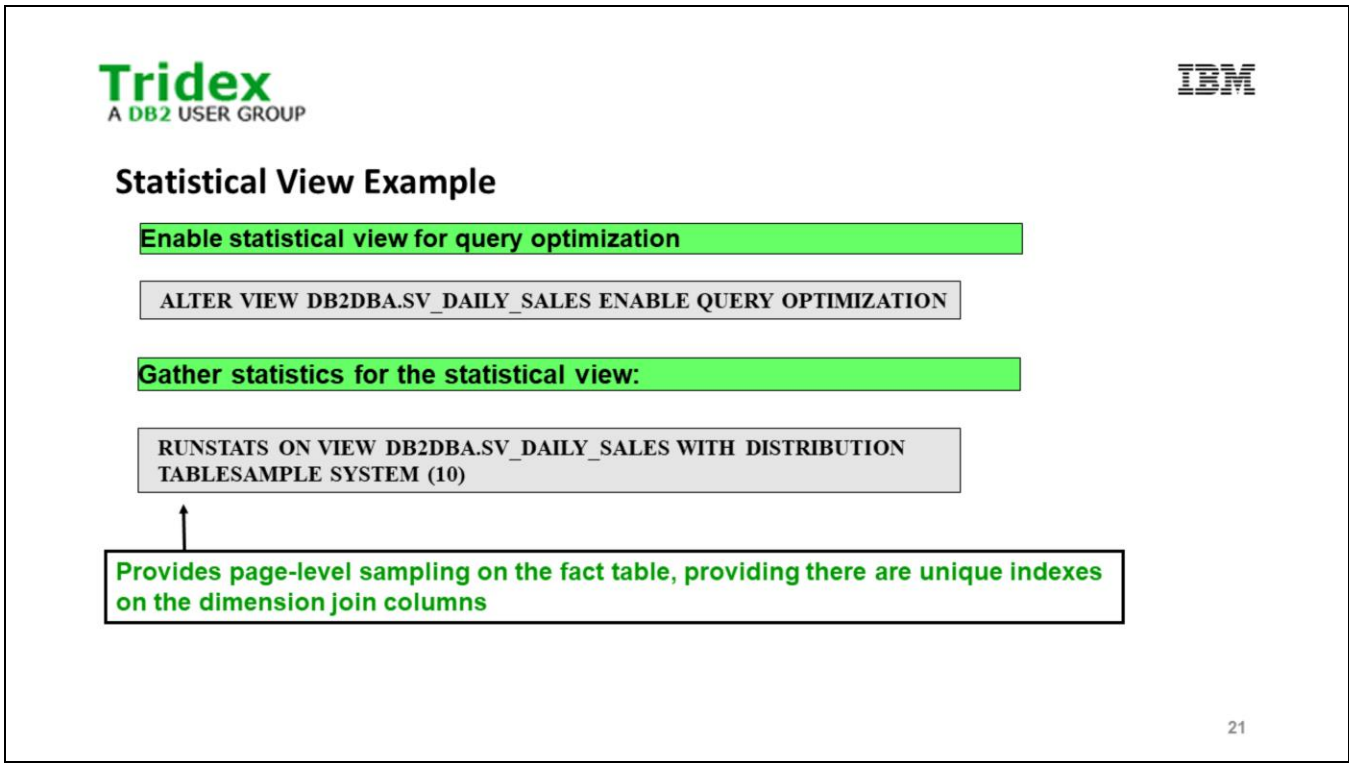
- Define RI constraint so that the statistical view can be used for any query that references the fact table and any subset of the dimension tables.
- RI can be unenforced or statistical
- "Statistical" means:
  - Parent-child relationship isn't perfect
  - No semantic query optimizations will occur

```
ALTER TABLE STORE ADD CONSTRAINT  FK1 FOREIGN KEY (STOREKEY)
REFERENCES DAILY_SALES NOT ENFORCED NOT TRUSTED

ALTER TABLE PERIOD etc.

ALTER TABLE PRODUCT etc.
```

20



Tridex
A DB2 USER GROUP

IBM

Statistical View Example

Enable statistical view for query optimization

```
ALTER VIEW DB2DBA.SV_DAILY_SALES ENABLE QUERY OPTIMIZATION
```

Gather statistics for the statistical view:

```
RUNSTATS ON VIEW DB2DBA.SV_DAILY_SALES WITH DISTRIBUTION
TABLESAMPLE SYSTEM (10)
```

Provides page-level sampling on the fact table, providing there are unique indexes on the dimension join columns

21

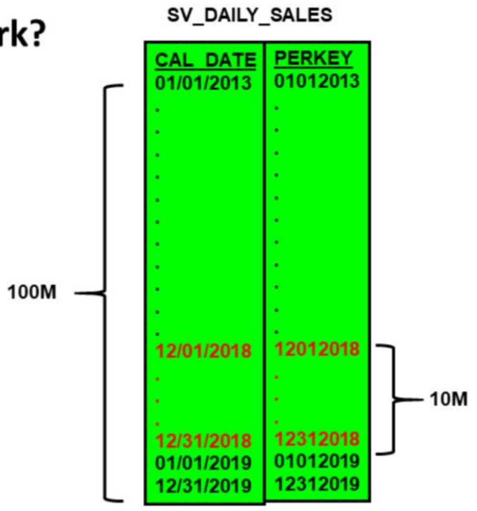
Tridex
A DB2 USER GROUP

IBM

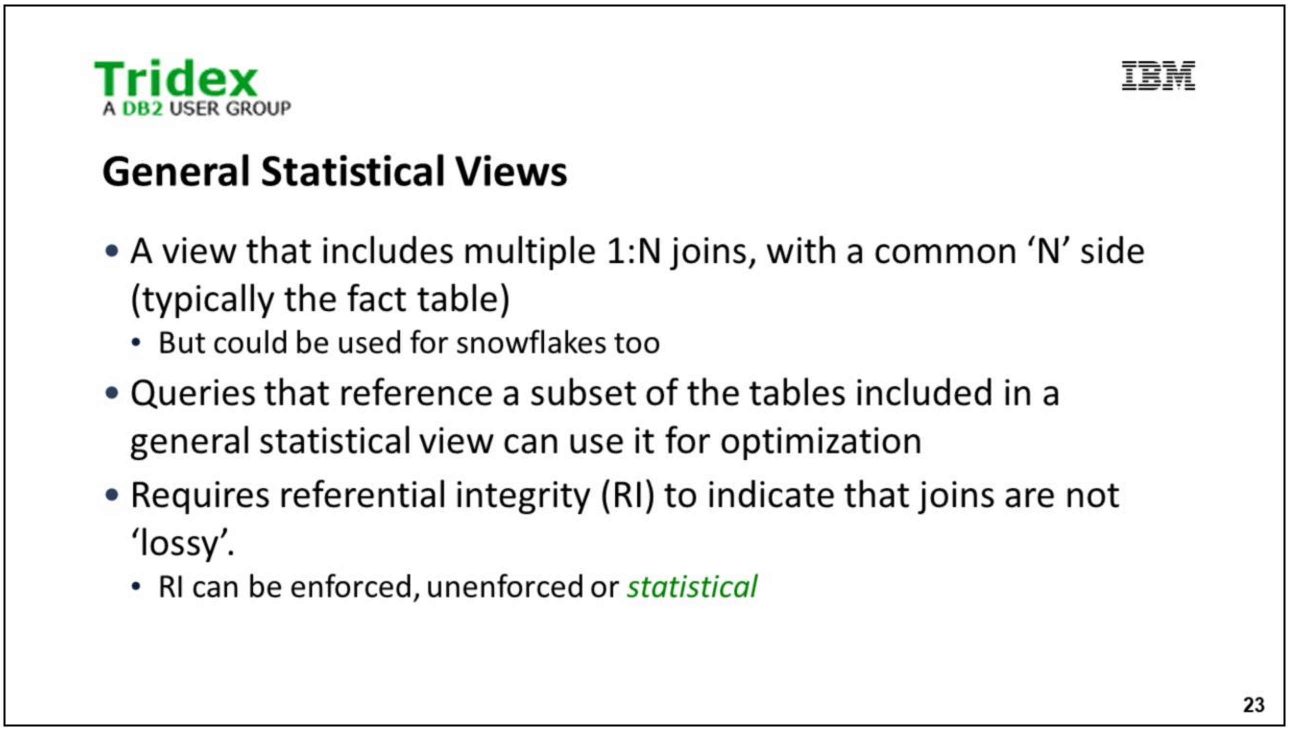
Statistical View Example – How Does it Work?

- SV_DAILY_SALES represents the statistics after joining PERIOD and DAILY_SALES
- SV_DAILY_SALES.CAL_DATE contains only 7 years of dates instead of 100
- The optimizer uses the statistics from SV_DAILY_SALES instead of PERIOD

```
SELECT …
FROM PERIOD, DAILY_SALES
WHERE
PERIOD.PERKEY=DAILY_SALES.PERKEY AND
CALENDAR_DATE BETWEEN '12/01/2018'
AND '12/31/2018' AND …
```



22


Tridex
A DB2 USER GROUP

IBM

General Statistical Views

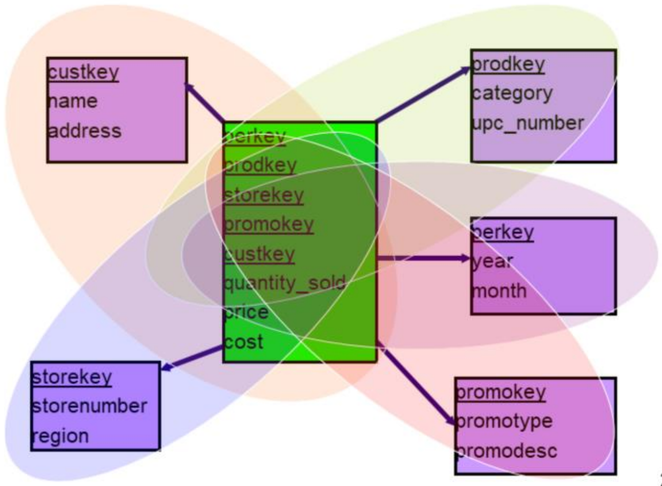
- A view that includes multiple 1:N joins, with a common 'N' side (typically the fact table)
  - But could be used for snowflakes too
- Queries that reference a subset of the tables included in a general statistical view can use it for optimization
- Requires referential integrity (RI) to indicate that joins are not 'lossy'.
  - RI can be enforced, unenforced or *statistical*

Tridex
A DB2 USER GROUP

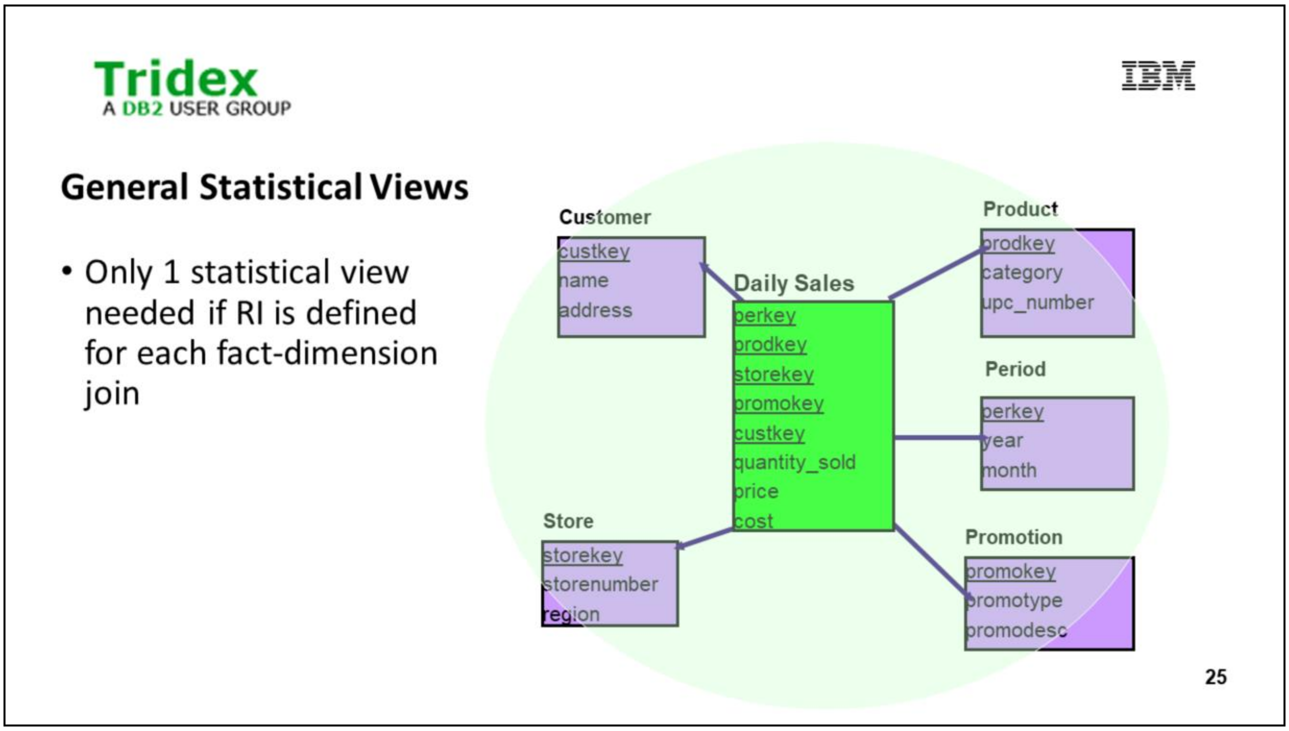
IBM

General Statistical Views

- Without RI, a separate statistical view is needed for every fact-dimension join



24

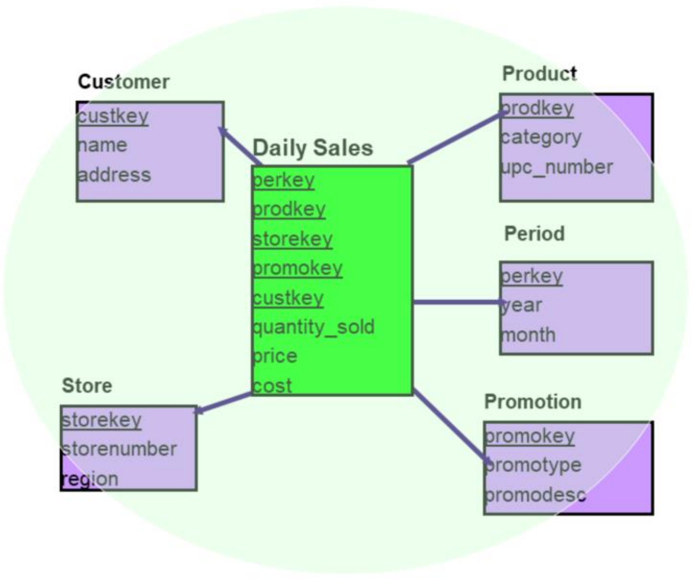


Tridex
A DB2 USER GROUP

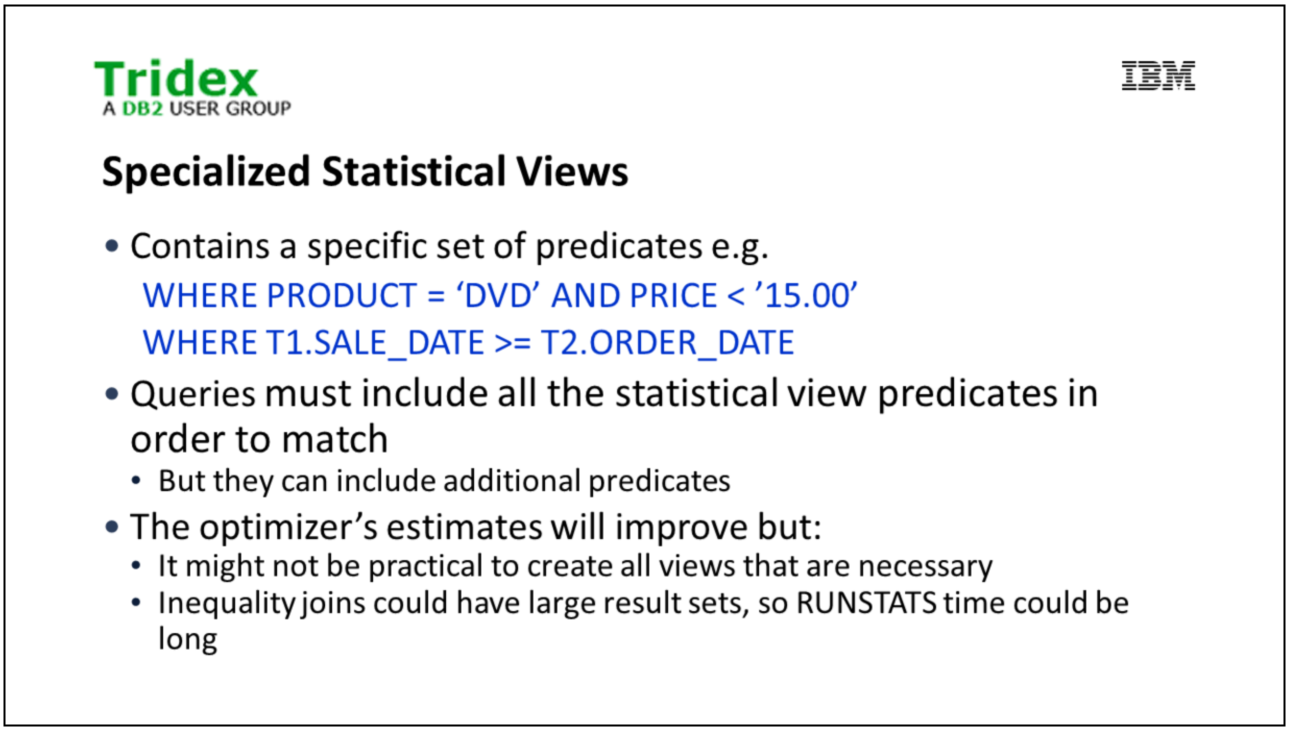
IBM

General Statistical Views

- Only 1 statistical view needed if RI is defined for each fact-dimension join



25



Tridex
A DB2 USER GROUP

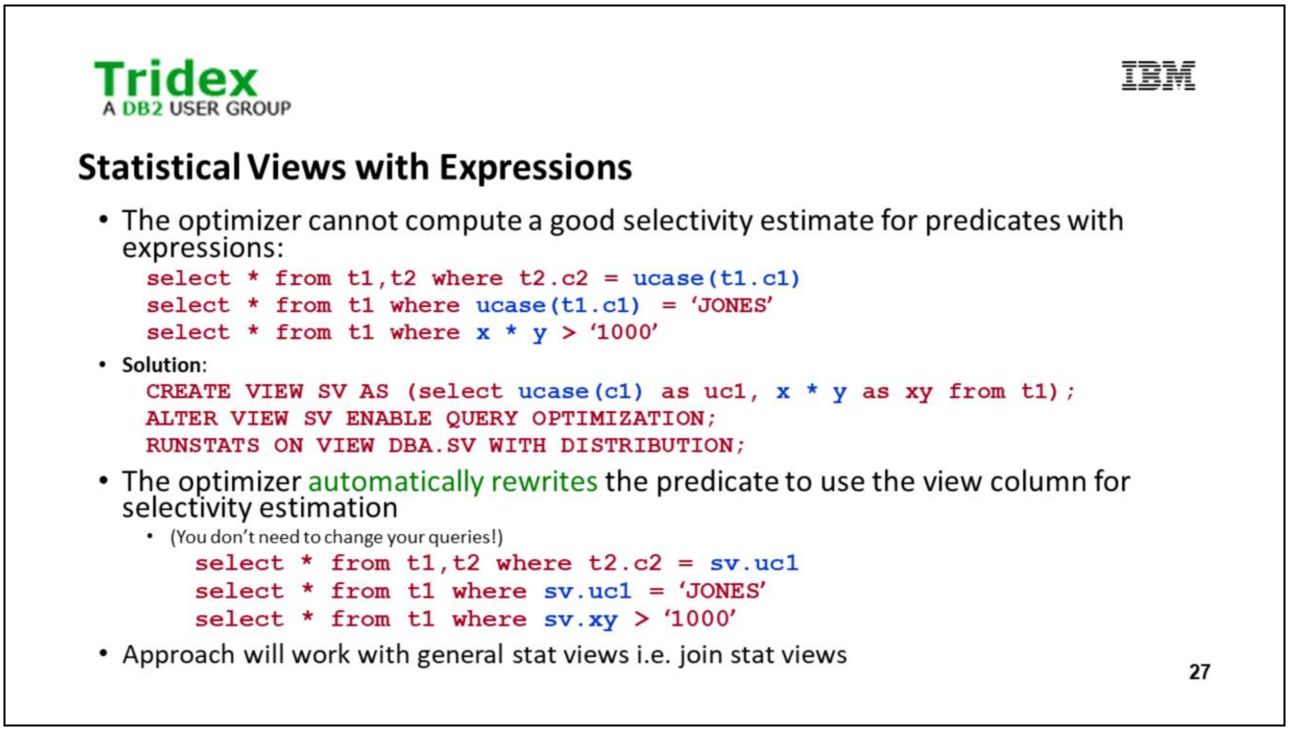
IBM

Specialized Statistical Views

- Contains a specific set of predicates e.g.

  WHERE PRODUCT = ‘DVD’ AND PRICE < ’15.00’

  WHERE T1.SALE_DATE >= T2.ORDER_DATE
- Queries must include all the statistical view predicates in order to match
  - But they can include additional predicates
- The optimizer’s estimates will improve but:
  - It might not be practical to create all views that are necessary
  - Inequality joins could have large result sets, so RUNSTATS time could be long



Statistical Views with Expressions

- The optimizer cannot compute a good selectivity estimate for predicates with expressions:

```
select * from t1,t2 where t2.c2 = ucase(t1.c1)
select * from t1 where ucase(t1.c1)  = 'JONES'
select * from t1 where x * y > '1000'
```

- **Solution**:

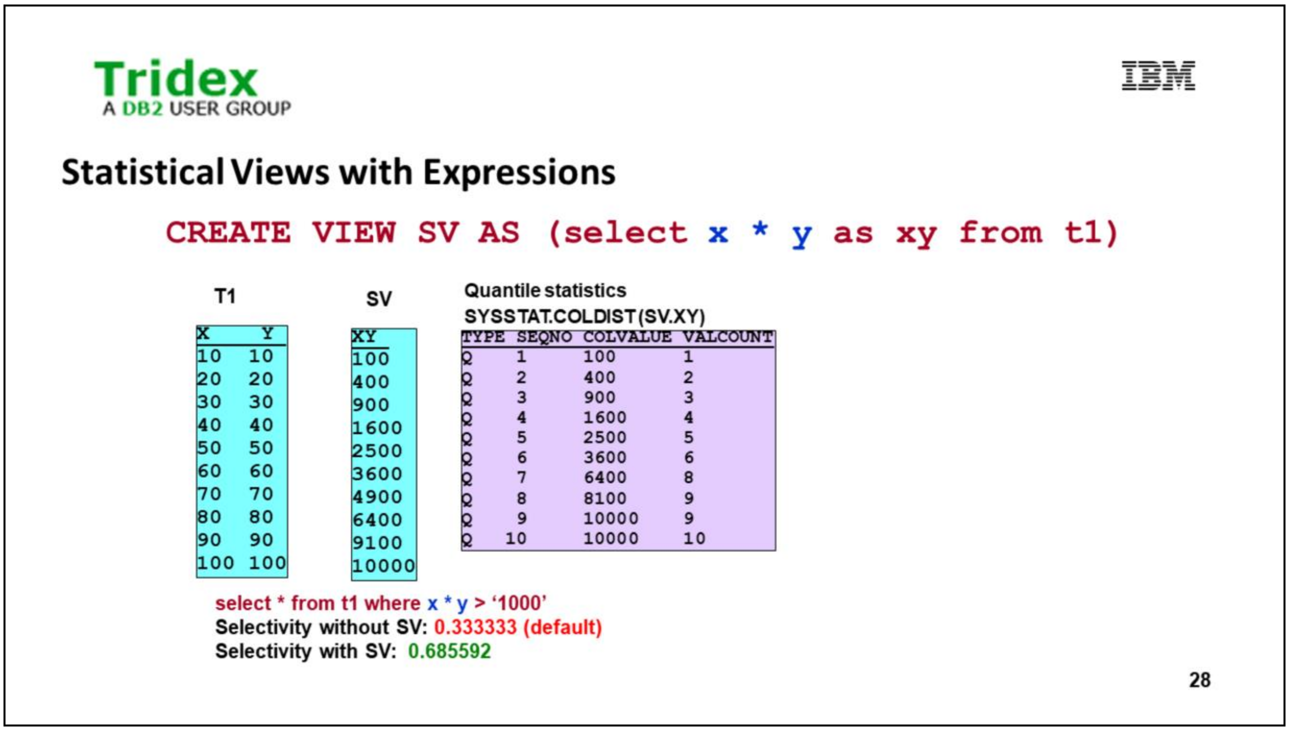
```
CREATE VIEW SV AS (select ucase(c1) as uc1, x * y as xy from t1);
ALTER VIEW SV ENABLE QUERY OPTIMIZATION;
RUNSTATS ON VIEW DBA.SV WITH DISTRIBUTION;
```

- The optimizer automatically rewrites the predicate to use the view column for selectivity estimation
  - (You don't need to change your queries!)

```
    select * from t1,t2 where t2.c2 = sv.uc1
    select * from t1 where sv.uc1 = 'JONES'
    select * from t1 where sv.xy > '1000'
```

- Approach will work with general stat views i.e. join stat views

27



Tridex
A DB2 USER GROUP

IBM

Statistical Views with Expressions

```
CREATE VIEW SV AS (select x * y as xy from t1)
```

T1

X	Y
10	10
20	20
30	30
40	40
50	50
60	60
70	70
80	80
90	90
100	100

SV

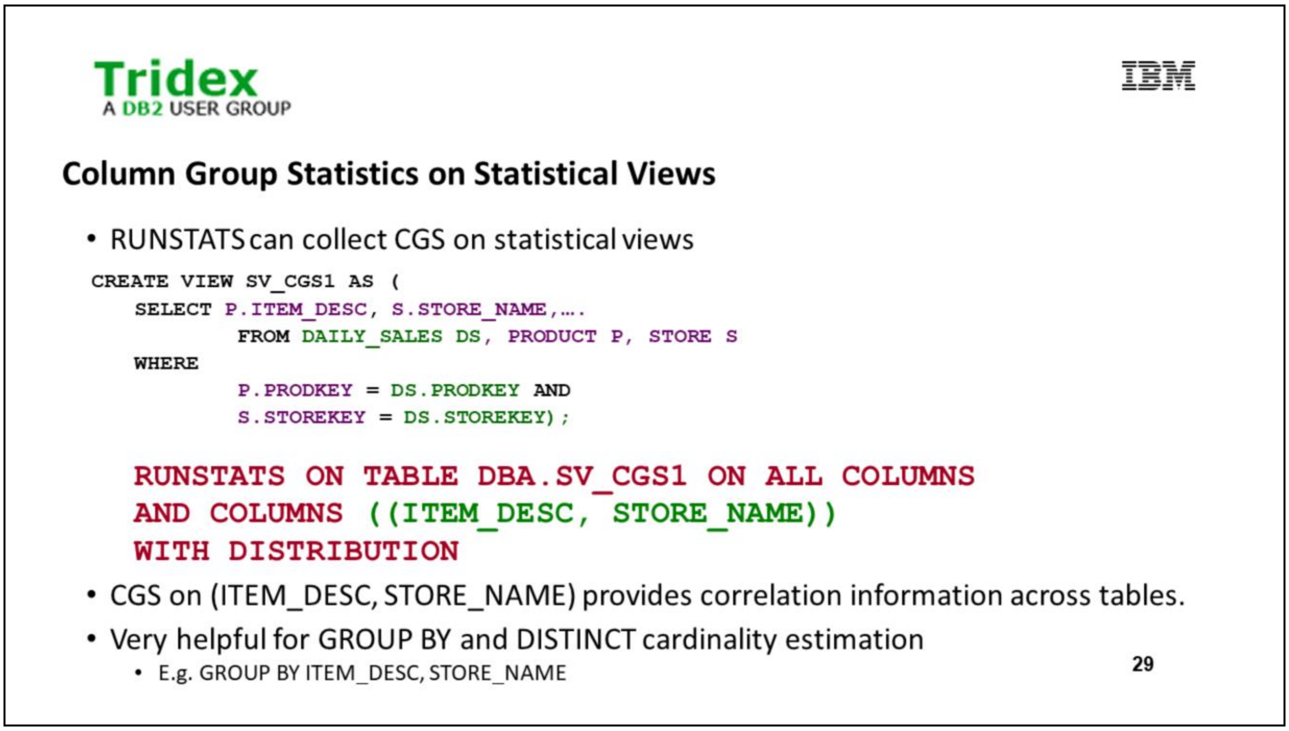
XY
100
400
900
1600
2500
3600
4900
6400
9100
10000

Quantile statistics
SYSSTAT.COLDIST(SV.XY)

TYPE	SEQNO	COLVALUE	VALCOUNT
Q	1	100	1
Q	2	400	2
Q	3	900	3
Q	4	1600	4
Q	5	2500	5
Q	6	3600	6
Q	7	6400	8
Q	8	8100	9
Q	9	10000	9
Q	10	10000	10

select * from t1 where x * y > '1000'
Selectivity without SV: 0.333333 (default)
Selectivity with SV: 0.685592

28



Column Group Statistics on Statistical Views

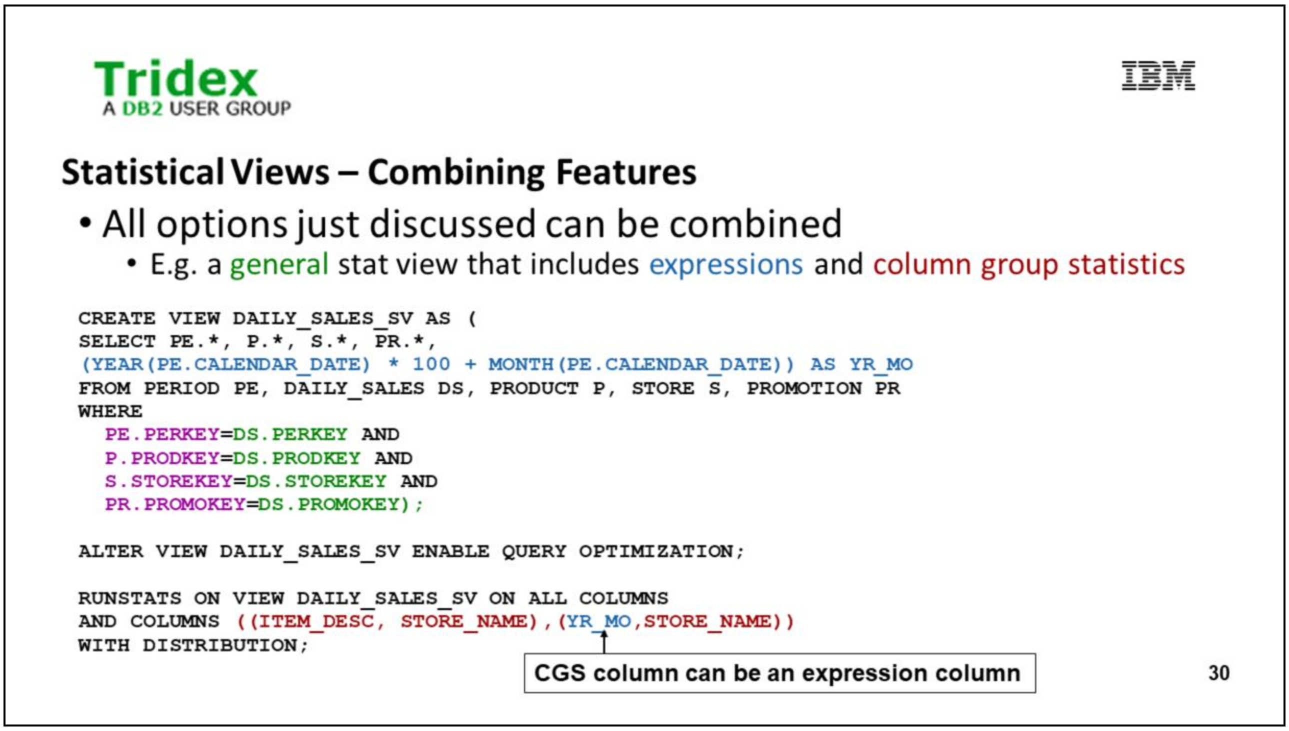
- RUNSTATS can collect CGS on statistical views

```
CREATE VIEW SV_CGS1 AS (
    SELECT P.ITEM_DESC, S.STORE_NAME,….
            FROM DAILY_SALES DS, PRODUCT P, STORE S
    WHERE
            P.PRODKEY = DS.PRODKEY AND
            S.STOREKEY = DS.STOREKEY);
```

```
RUNSTATS ON TABLE DBA.SV_CGS1 ON ALL COLUMNS
AND COLUMNS ((ITEM_DESC, STORE_NAME))
WITH DISTRIBUTION
```

- CGS on (ITEM_DESC, STORE_NAME) provides correlation information across tables.
- Very helpful for GROUP BY and DISTINCT cardinality estimation
  - E.g. GROUP BY ITEM_DESC, STORE_NAME

29



Tridex
A DB2 USER GROUP

IBM

Statistical Views – Combining Features

- All options just discussed can be combined
  - E.g. a general stat view that includes expressions and column group statistics

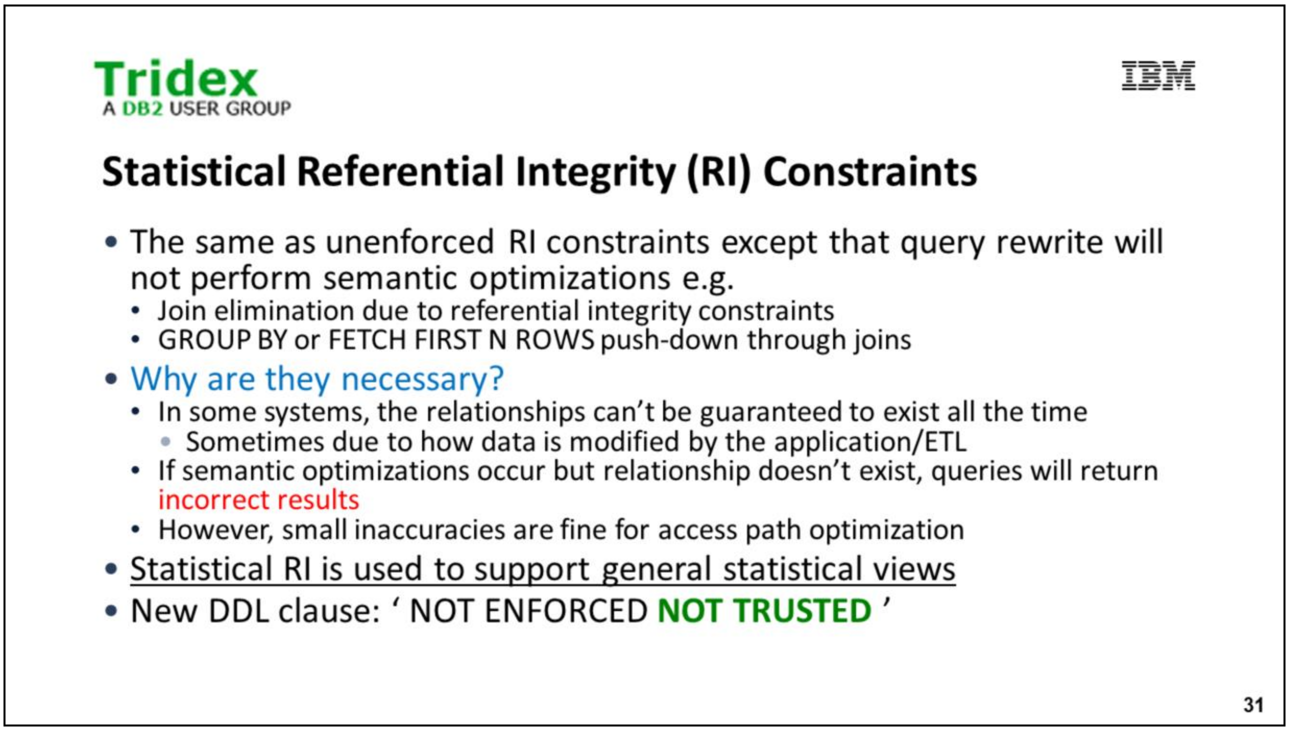
```
CREATE VIEW DAILY_SALES_SV AS (
SELECT PE.*, P.*, S.*, PR.*,
(YEAR(PE.CALENDAR_DATE) * 100 + MONTH(PE.CALENDAR_DATE)) AS YR_MO
FROM PERIOD PE, DAILY_SALES DS, PRODUCT P, STORE S, PROMOTION PR
WHERE
  PE.PERKEY=DS.PERKEY AND
  P.PRODKEY=DS.PRODKEY AND
  S.STOREKEY=DS.STOREKEY AND
  PR.PROMOKEY=DS.PROMOKEY);

ALTER VIEW DAILY_SALES_SV ENABLE QUERY OPTIMIZATION;

RUNSTATS ON VIEW DAILY_SALES_SV ON ALL COLUMNS
AND COLUMNS ((ITEM_DESC, STORE_NAME),(YR_MO,STORE_NAME))
WITH DISTRIBUTION;
```

CGS column can be an expression column

30



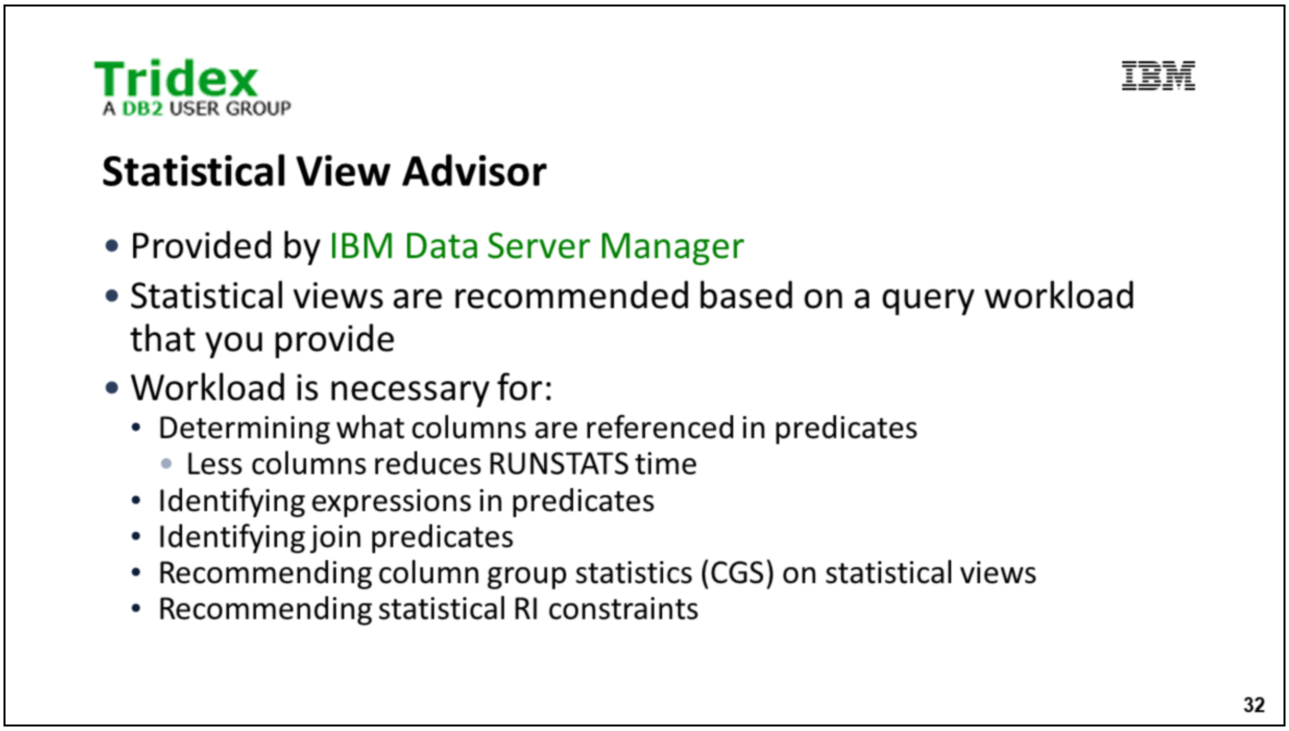
Tridex
A DB2 USER GROUP

IBM

Statistical Referential Integrity (RI) Constraints

- The same as unenforced RI constraints except that query rewrite will not perform semantic optimizations e.g.
  - Join elimination due to referential integrity constraints
  - GROUP BY or FETCH FIRST N ROWS push-down through joins
- Why are they necessary?
  - In some systems, the relationships can't be guaranteed to exist all the time
    - Sometimes due to how data is modified by the application/ETL
  - If semantic optimizations occur but relationship doesn't exist, queries will return incorrect results
  - However, small inaccuracies are fine for access path optimization
- Statistical RI is used to support general statistical views
- New DDL clause: ' NOT ENFORCED **NOT TRUSTED** '

31



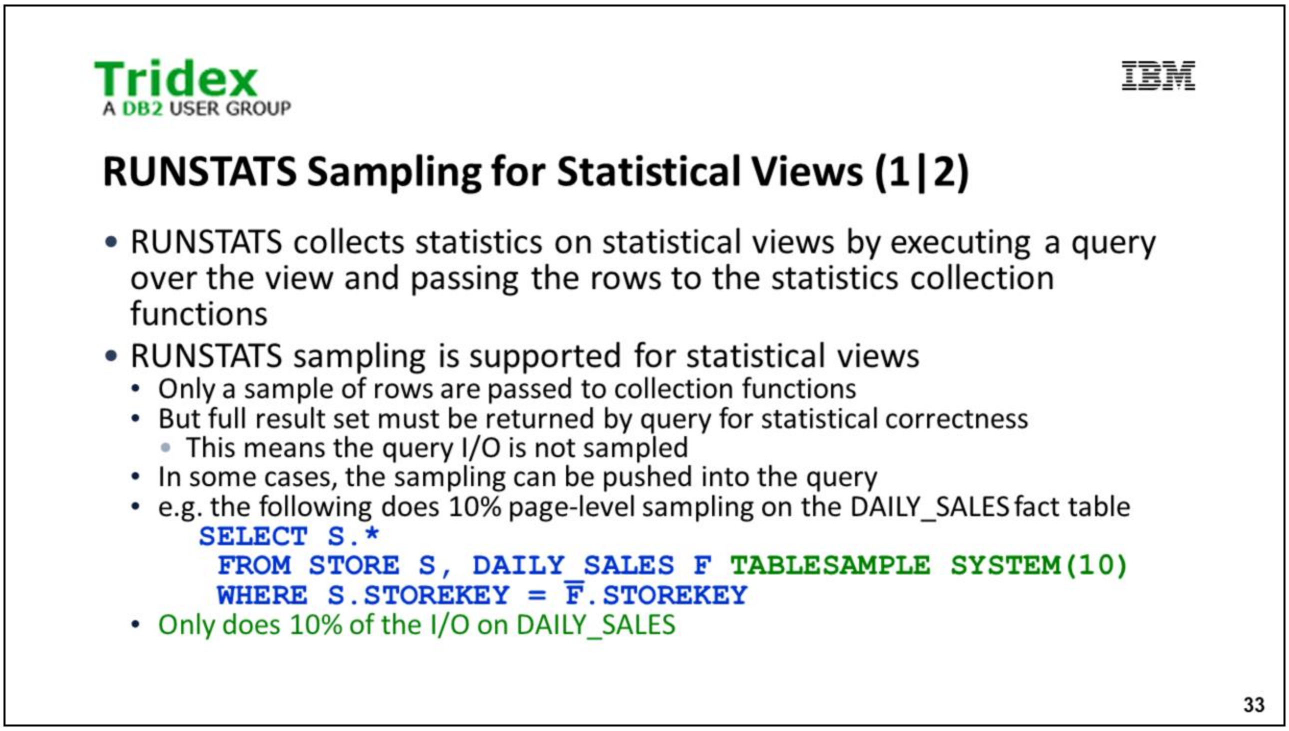
Tridex
A DB2 USER GROUP

IBM

Statistical View Advisor

- Provided by IBM Data Server Manager
- Statistical views are recommended based on a query workload that you provide
- Workload is necessary for:
  - Determining what columns are referenced in predicates
    - Less columns reduces RUNSTATS time
  - Identifying expressions in predicates
  - Identifying join predicates
  - Recommending column group statistics (CGS) on statistical views
  - Recommending statistical RI constraints

32



Tridex
A DB2 USER GROUP

IBM

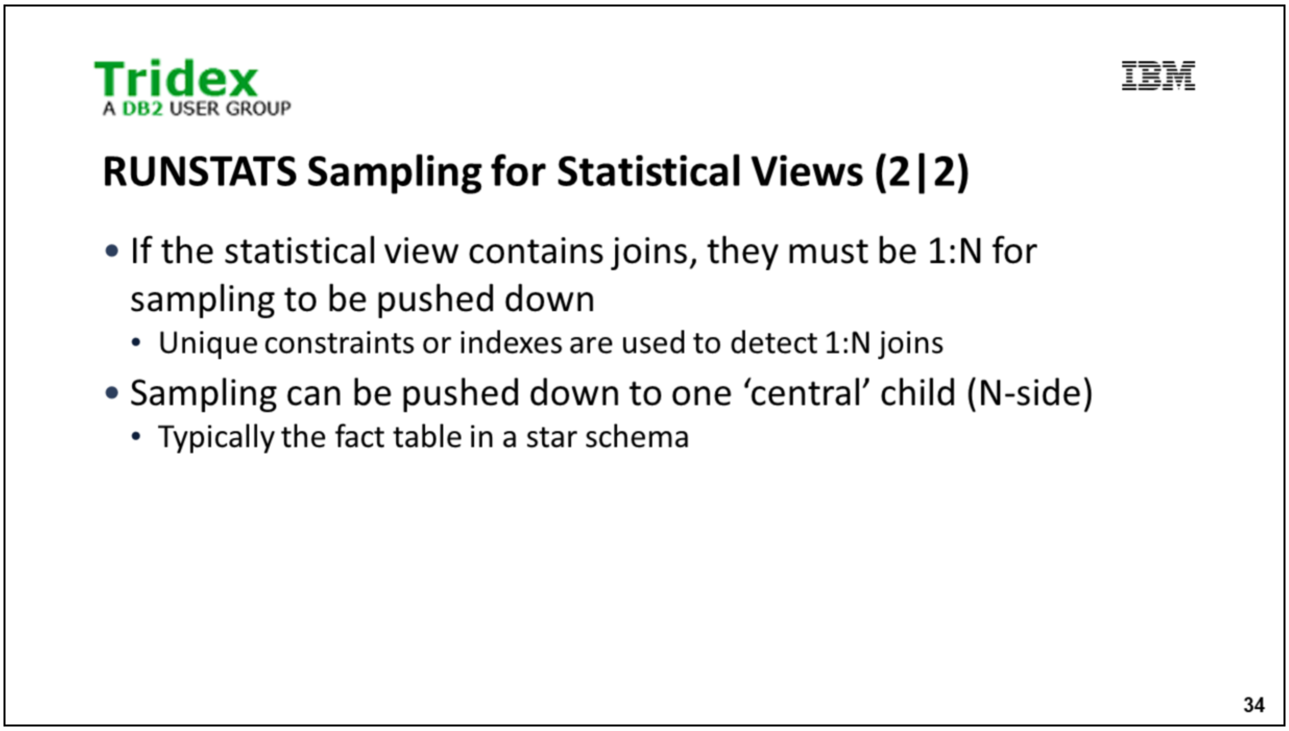
RUNSTATS Sampling for Statistical Views (1|2)

- RUNSTATS collects statistics on statistical views by executing a query over the view and passing the rows to the statistics collection functions
- RUNSTATS sampling is supported for statistical views
  - Only a sample of rows are passed to collection functions
  - But full result set must be returned by query for statistical correctness
    - This means the query I/O is not sampled
  - In some cases, the sampling can be pushed into the query
  - e.g. the following does 10% page-level sampling on the DAILY_SALES fact table

```
SELECT S.*
 FROM STORE S, DAILY_SALES F TABLESAMPLE SYSTEM(10)
 WHERE S.STOREKEY = F.STOREKEY
```

  - Only does 10% of the I/O on DAILY_SALES

33



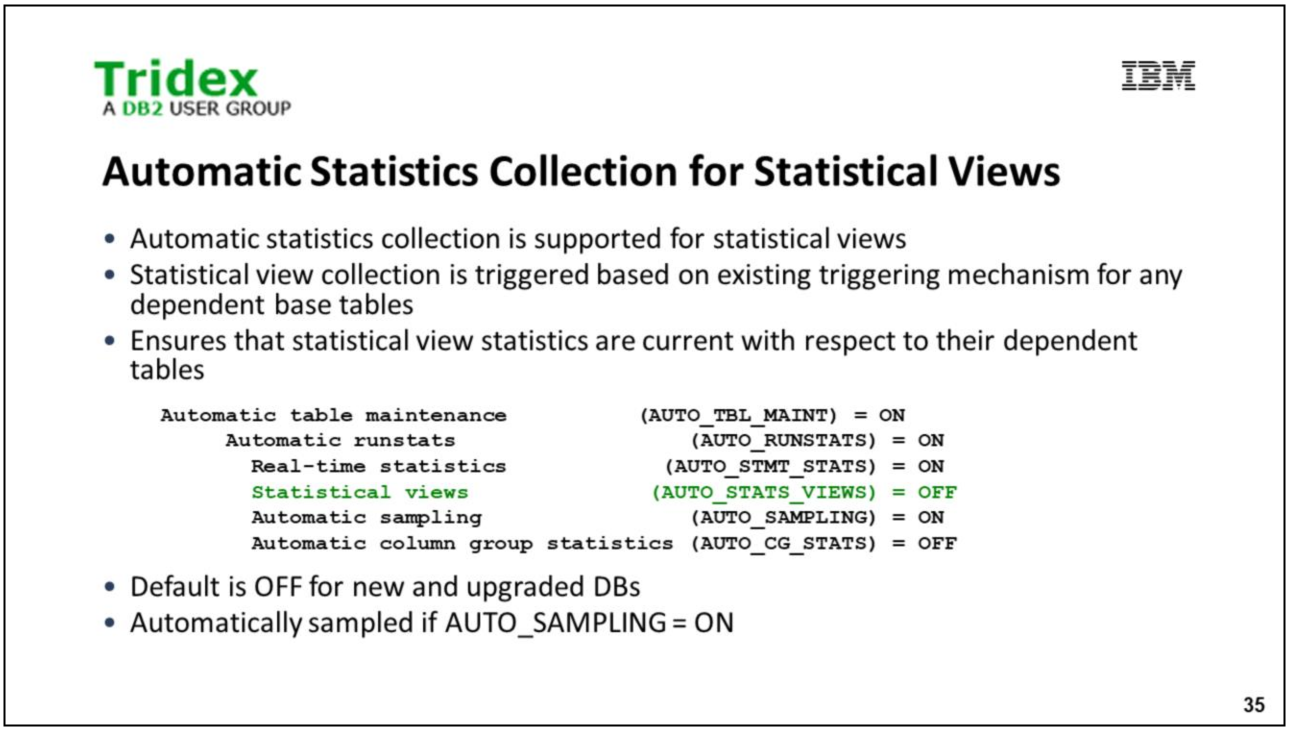
Tridex
A DB2 USER GROUP

IBM

RUNSTATS Sampling for Statistical Views (2|2)

- If the statistical view contains joins, they must be 1:N for sampling to be pushed down
  - Unique constraints or indexes are used to detect 1:N joins
- Sampling can be pushed down to one 'central' child (N-side)
  - Typically the fact table in a star schema

34



Tridex
A DB2 USER GROUP

IBM

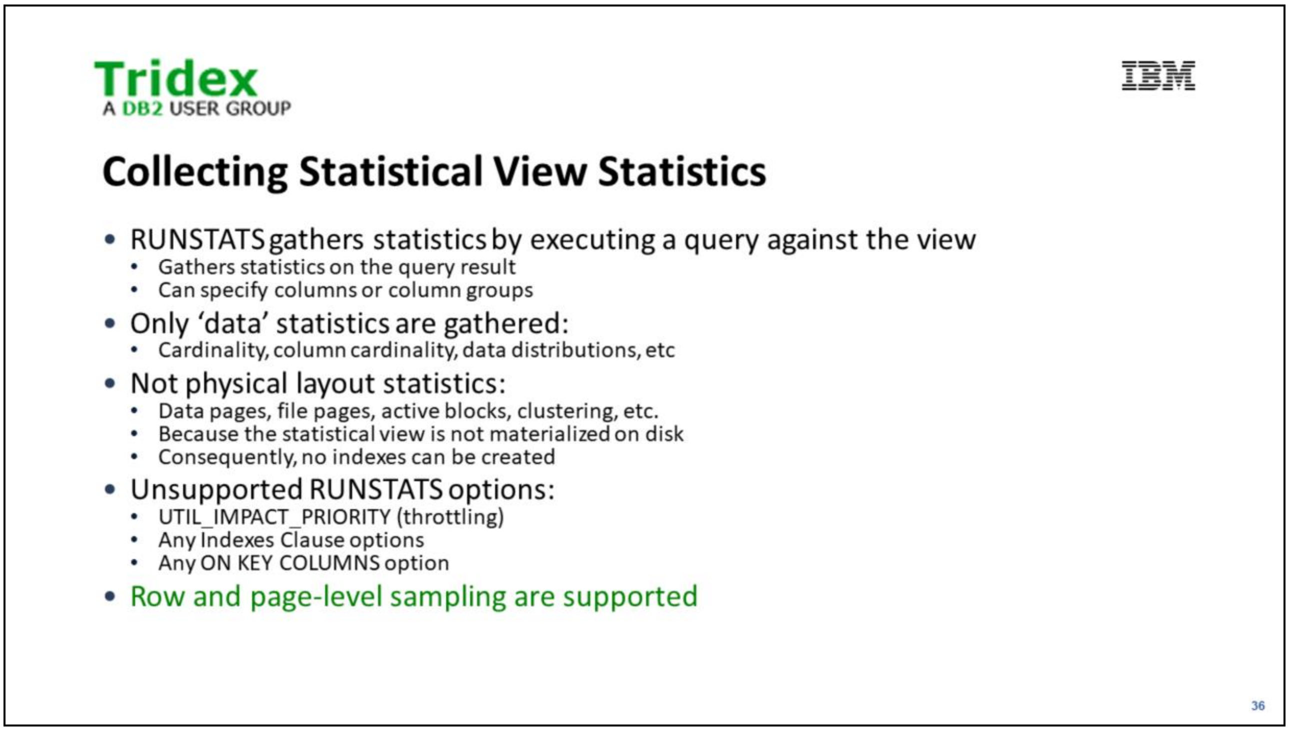
Automatic Statistics Collection for Statistical Views

- Automatic statistics collection is supported for statistical views
- Statistical view collection is triggered based on existing triggering mechanism for any dependent base tables
- Ensures that statistical view statistics are current with respect to their dependent tables

```
Automatic table maintenance          (AUTO_TBL_MAINT) = ON
     Automatic runstats                  (AUTO_RUNSTATS) = ON
       Real-time statistics            (AUTO_STMT_STATS) = ON
       Statistical views              (AUTO_STATS_VIEWS) = OFF
       Automatic sampling                (AUTO_SAMPLING) = ON
       Automatic column group statistics (AUTO_CG_STATS) = OFF
```

- Default is OFF for new and upgraded DBs
- Automatically sampled if AUTO_SAMPLING = ON

35



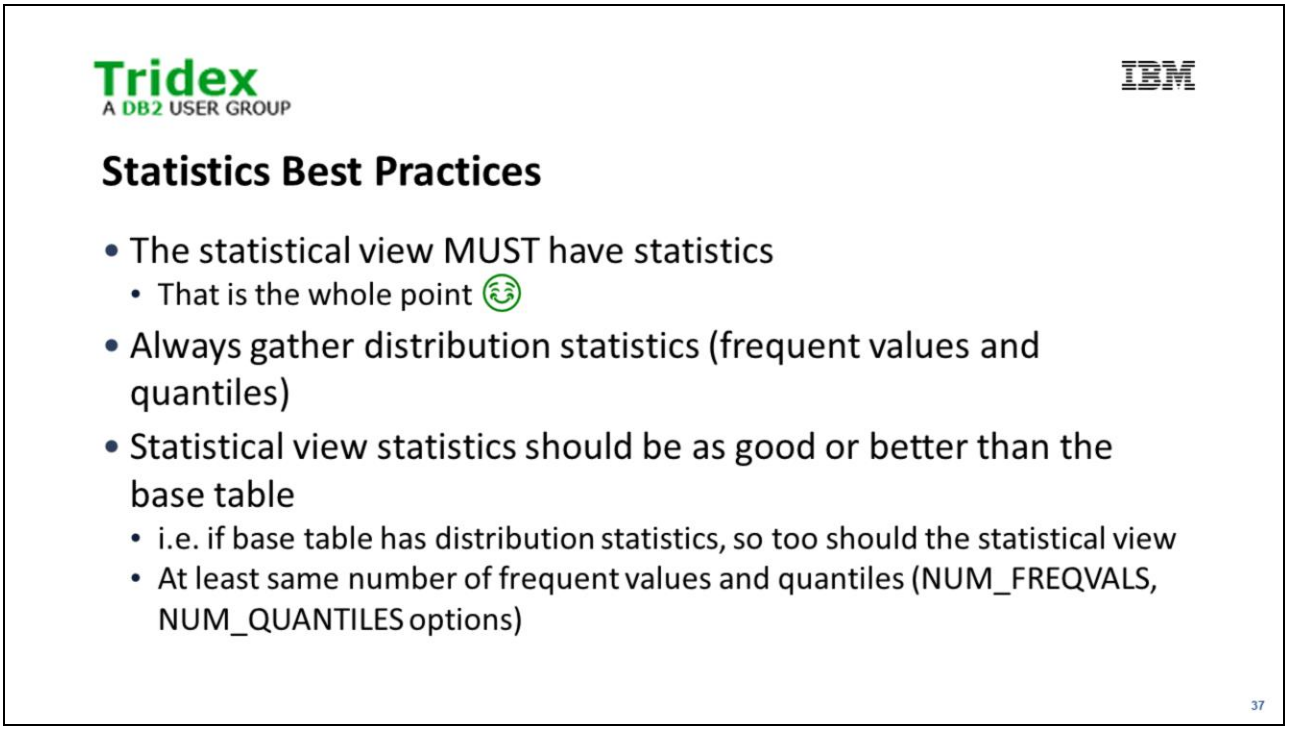
Tridex
A DB2 USER GROUP

IBM

Collecting Statistical View Statistics

- RUNSTATS gathers statistics by executing a query against the view
  - Gathers statistics on the query result
  - Can specify columns or column groups
- Only 'data' statistics are gathered:
  - Cardinality, column cardinality, data distributions, etc
- Not physical layout statistics:
  - Data pages, file pages, active blocks, clustering, etc.
  - Because the statistical view is not materialized on disk
  - Consequently, no indexes can be created
- Unsupported RUNSTATS options:
  - UTIL_IMPACT_PRIORITY (throttling)
  - Any Indexes Clause options
  - Any ON KEY COLUMNS option
- Row and page-level sampling are supported

36



Tridex
A DB2 USER GROUP

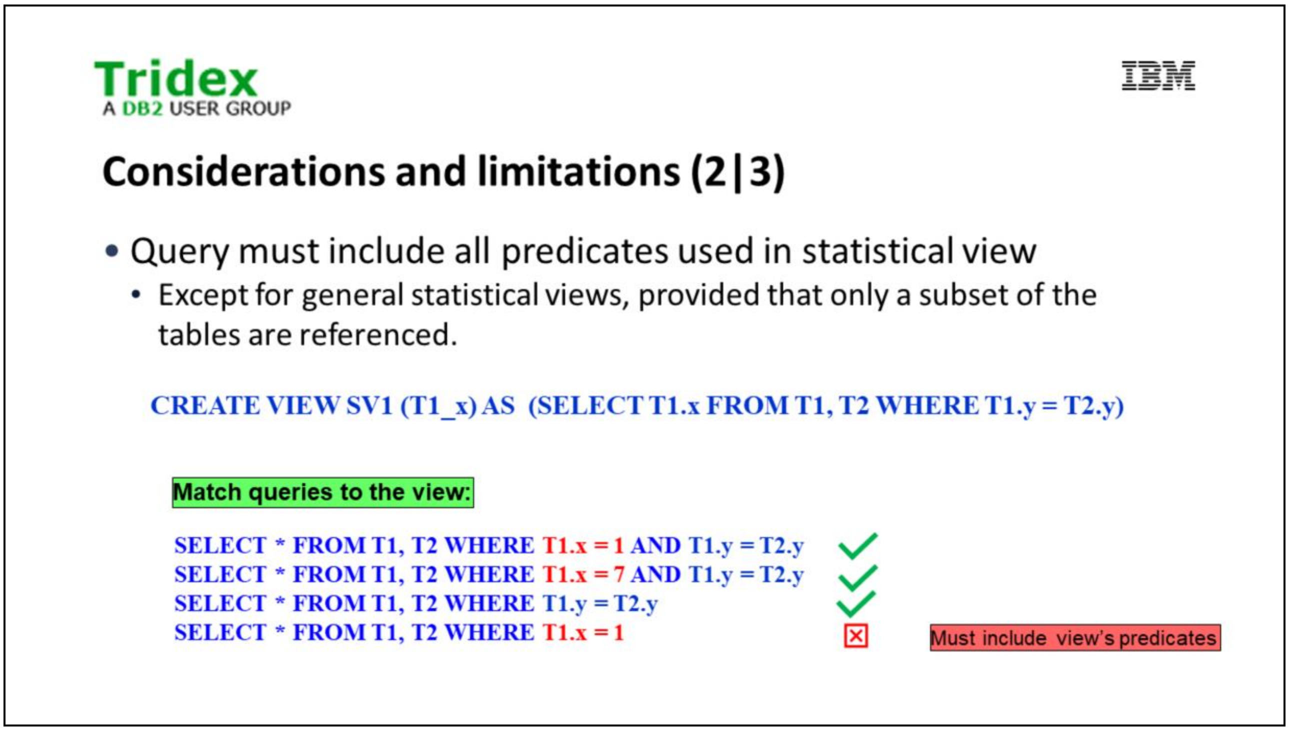
IBM

Statistics Best Practices

- The statistical view MUST have statistics
  - That is the whole point 😊
- Always gather distribution statistics (frequent values and quantiles)
- Statistical view statistics should be as good or better than the base table
  - i.e. if base table has distribution statistics, so too should the statistical view
  - At least same number of frequent values and quantiles (NUM_FREQVALS, NUM_QUANTILES options)

Considerations and limitations (1|3)

- Considerations:
  - Statistical view cannot contain:
    - Aggregation or distinct operations
    - UNION, EXCEPT, or INTERSECT operations
    - Scalar aggregate (OLAP) functions
    - Outer join
    - Warning returned during ALTER VIEW
      - SQL20278W The view "<viewname>" may not be used to optimize the processing of queries
  - Statistical views have many of the same matching restrictions as MQTs.
  - Use Explain diagnostic facility to understand why a statistical view or MQT may not have been used



Tridex
A DB2 USER GROUP

IBM

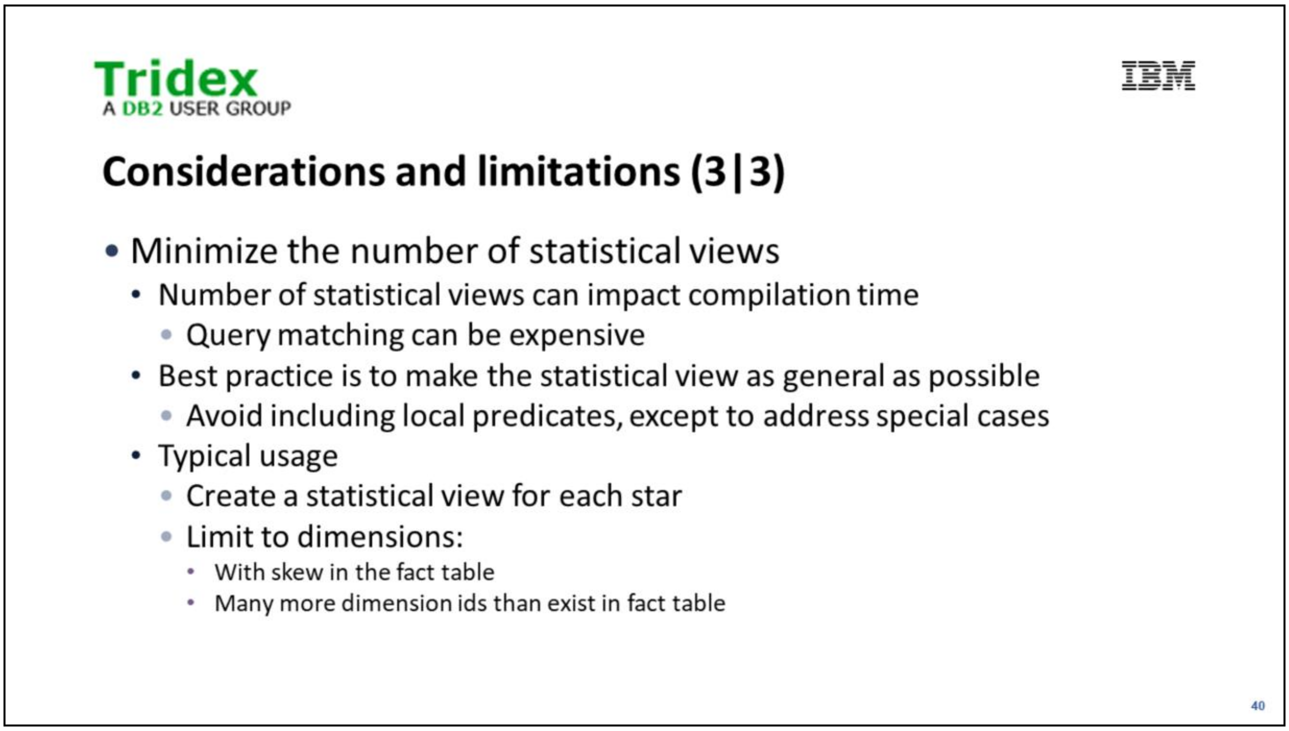
Considerations and limitations (2|3)

- Query must include all predicates used in statistical view
  - Except for general statistical views, provided that only a subset of the tables are referenced.

```
CREATE VIEW SV1 (T1_x) AS  (SELECT T1.x FROM T1, T2 WHERE T1.y = T2.y)
```

Match queries to the view:

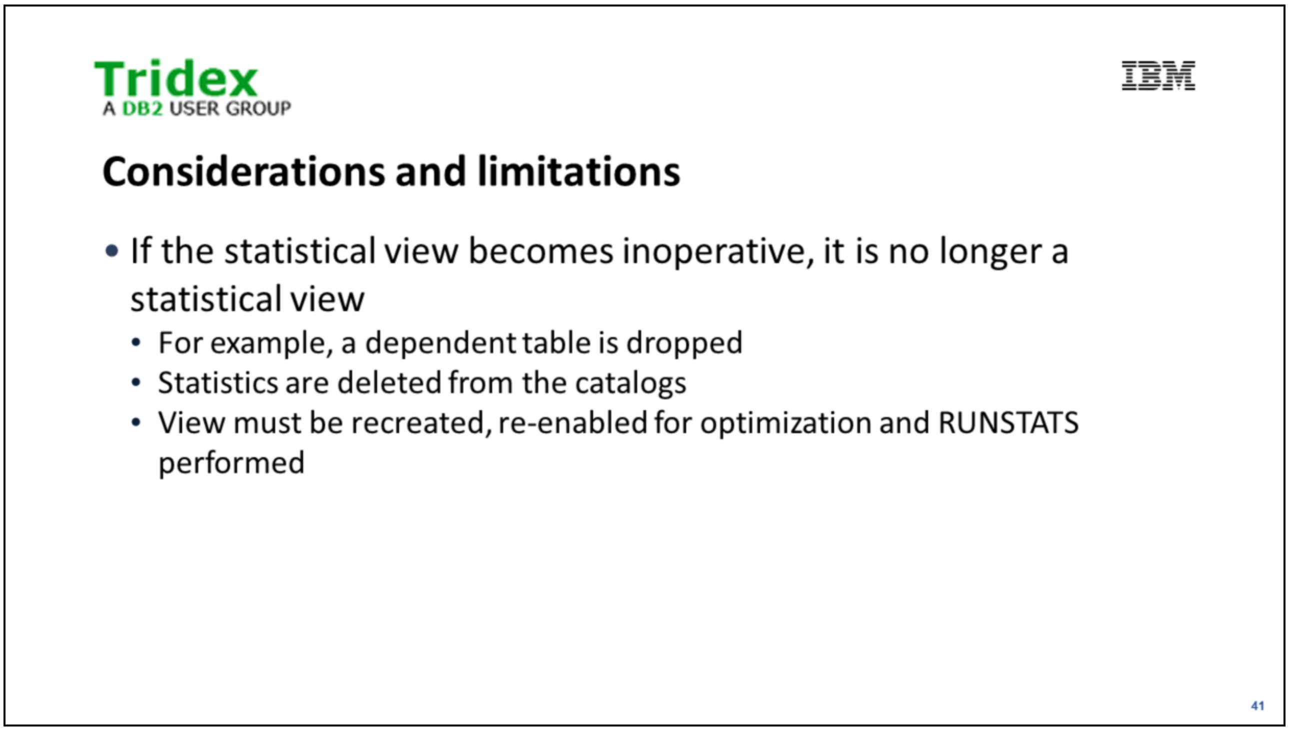
```
SELECT * FROM T1, T2 WHERE T1.x = 1 AND T1.y = T2.y     ✓
SELECT * FROM T1, T2 WHERE T1.x = 7 AND T1.y = T2.y     ✓
SELECT * FROM T1, T2 WHERE T1.y = T2.y                  ✓
SELECT * FROM T1, T2 WHERE T1.x = 1                     ☒     Must include  view's predicates
```



Considerations and limitations (3|3)

- Minimize the number of statistical views
  - Number of statistical views can impact compilation time
    - Query matching can be expensive
  - Best practice is to make the statistical view as general as possible
    - Avoid including local predicates, except to address special cases
  - Typical usage
    - Create a statistical view for each star
    - Limit to dimensions:
      - With skew in the fact table
      - Many more dimension ids than exist in fact table

40

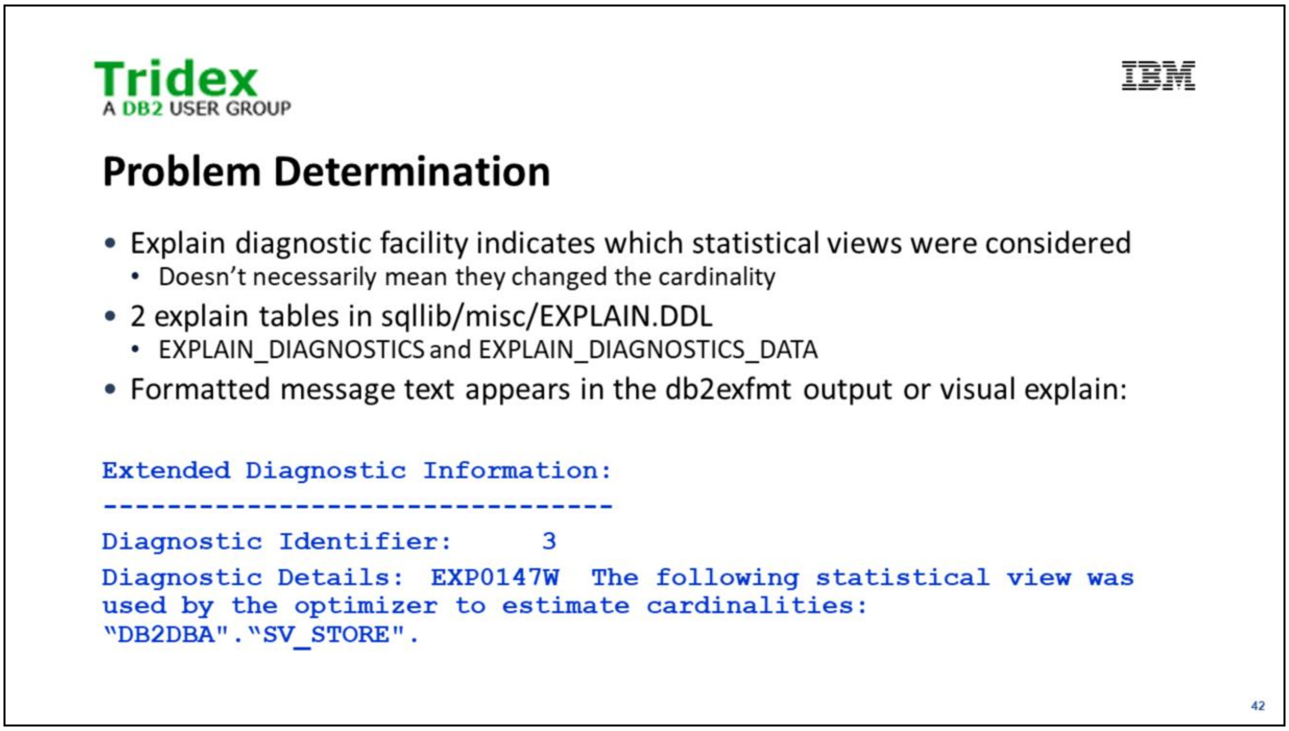


Tridex
A DB2 USER GROUP

IBM

Considerations and limitations

- If the statistical view becomes inoperative, it is no longer a statistical view
  - For example, a dependent table is dropped
  - Statistics are deleted from the catalogs
  - View must be recreated, re-enabled for optimization and RUNSTATS performed



Tridex
A DB2 USER GROUP

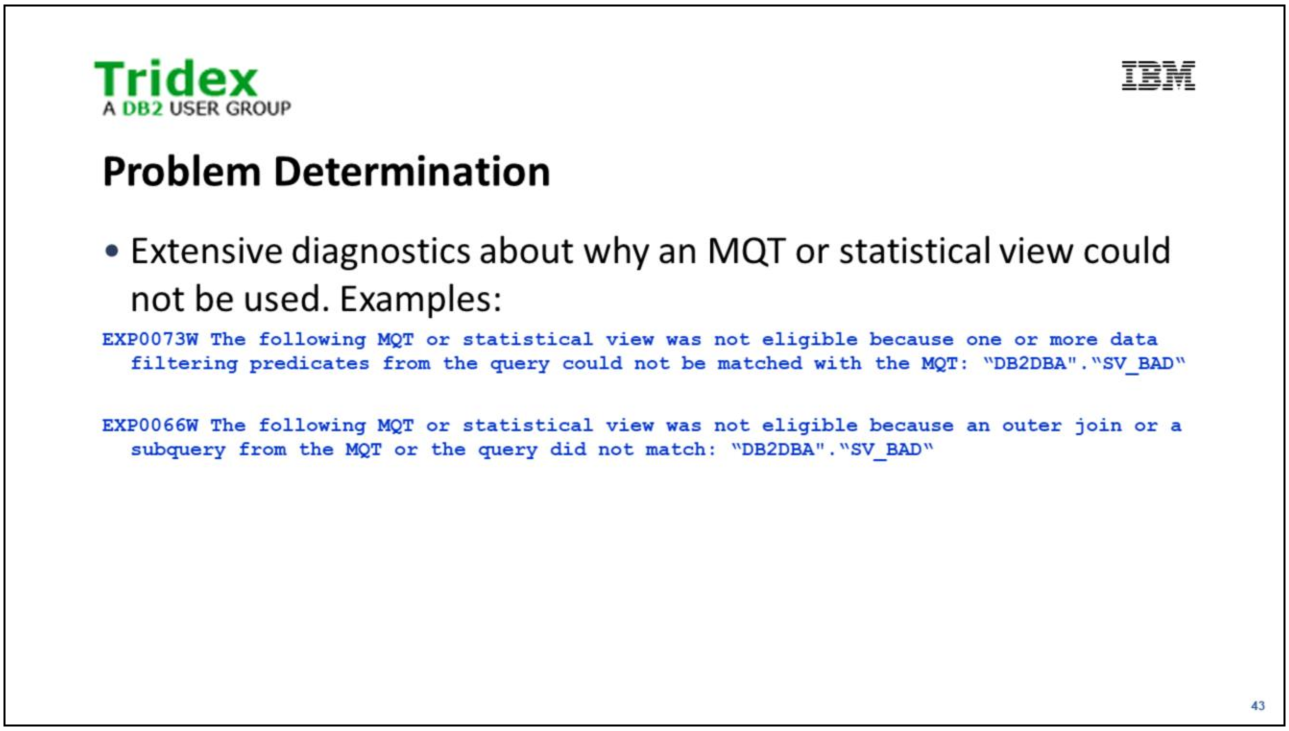
IBM

Problem Determination

- Explain diagnostic facility indicates which statistical views were considered
  - Doesn't necessarily mean they changed the cardinality
- 2 explain tables in sqllib/misc/EXPLAIN.DDL
  - EXPLAIN_DIAGNOSTICS and EXPLAIN_DIAGNOSTICS_DATA
- Formatted message text appears in the db2exfmt output or visual explain:

```
Extended Diagnostic Information:
--------------------------------
Diagnostic Identifier:     3
Diagnostic Details:  EXP0147W  The following statistical view was
used by the optimizer to estimate cardinalities:
"DB2DBA"."SV_STORE".
```

42


Tridex
A DB2 USER GROUP

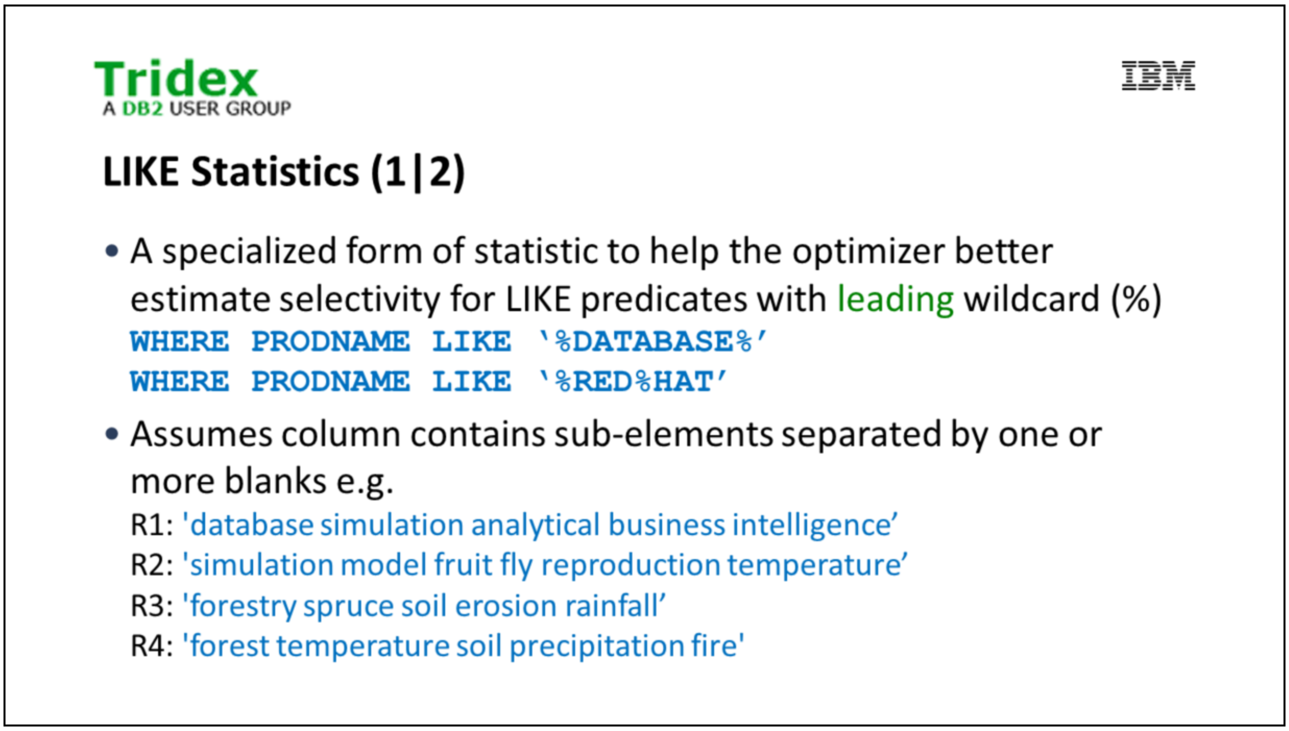
Problem Determination

- Extensive diagnostics about why an MQT or statistical view could not be used. Examples:

```
EXP0073W The following MQT or statistical view was not eligible because one or more data
  filtering predicates from the query could not be matched with the MQT: "DB2DBA"."SV_BAD"

EXP0066W The following MQT or statistical view was not eligible because an outer join or a
  subquery from the MQT or the query did not match: "DB2DBA"."SV_BAD"
```

43



Tridex
A DB2 USER GROUP

IBM

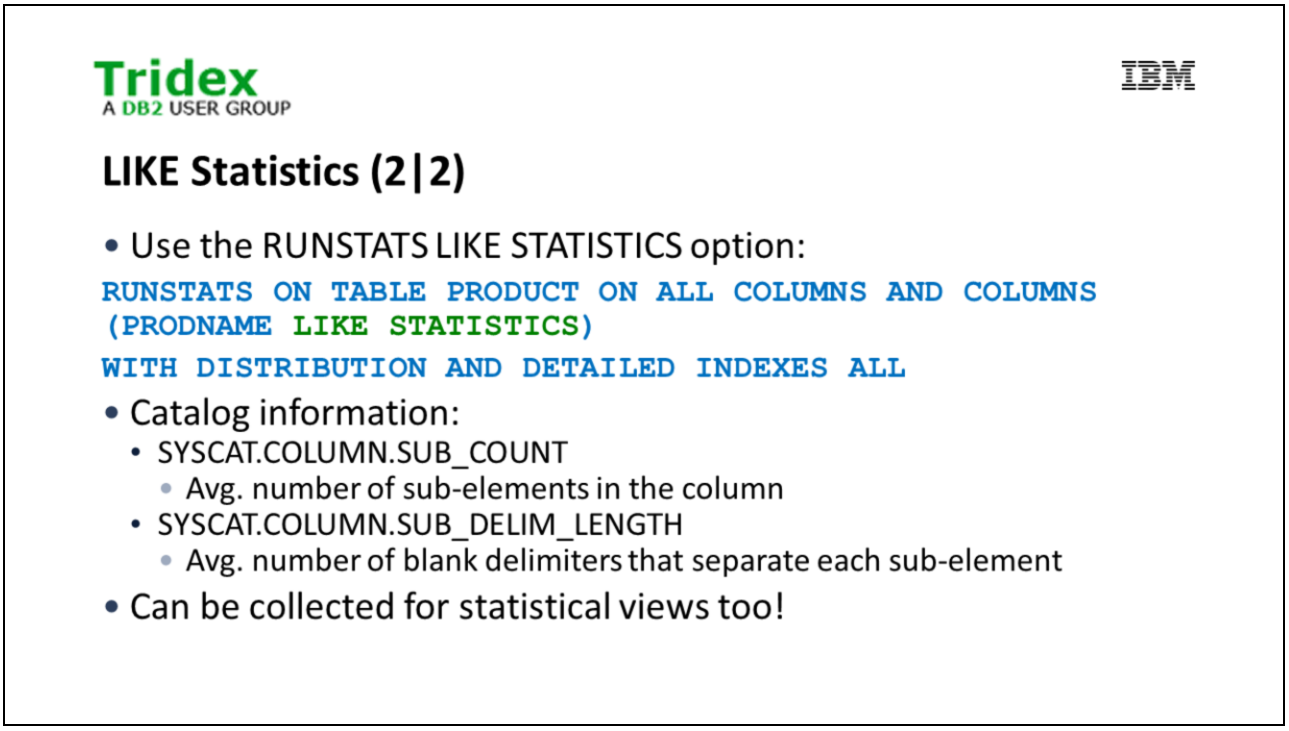
LIKE Statistics (1|2)

- A specialized form of statistic to help the optimizer better estimate selectivity for LIKE predicates with leading wildcard (%)

```
WHERE PRODNAME LIKE '%DATABASE%'
WHERE PRODNAME LIKE '%RED%HAT'
```

- Assumes column contains sub-elements separated by one or more blanks e.g.

  R1: 'database simulation analytical business intelligence'
  R2: 'simulation model fruit fly reproduction temperature'
  R3: 'forestry spruce soil erosion rainfall'
  R4: 'forest temperature soil precipitation fire'


Tridex
A DB2 USER GROUP

IBM

LIKE Statistics (2|2)

- Use the RUNSTATS LIKE STATISTICS option:

```
RUNSTATS ON TABLE PRODUCT ON ALL COLUMNS AND COLUMNS
(PRODNAME LIKE STATISTICS)
WITH DISTRIBUTION AND DETAILED INDEXES ALL
```

- Catalog information:
  - SYSCAT.COLUMN.SUB_COUNT
    - Avg. number of sub-elements in the column
  - SYSCAT.COLUMN.SUB_DELIM_LENGTH
    - Avg. number of blank delimiters that separate each sub-element
- Can be collected for statistical views too!

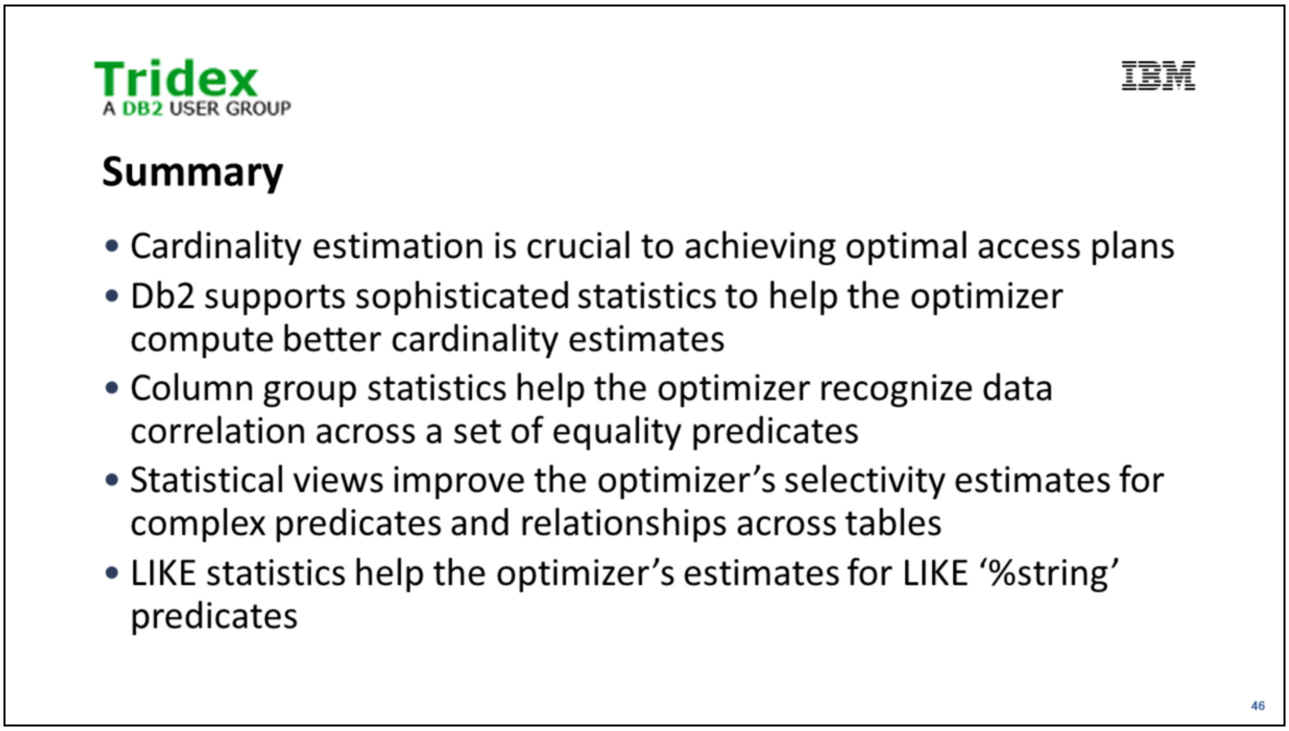
For the following 4 rows of data, SUB_COUNT is 5 and SUB_DELIM_LENGTH is 1

R1: 'database simulation analytical business intelligence’

R2: 'simulation model fruit fly reproduction temperature’

R3: 'forestry spruce soil erosion rainfall’

R4: 'forest temperature soil precipitation fire'



Tridex
A DB2 USER GROUP

IBM

Summary

- Cardinality estimation is crucial to achieving optimal access plans
- Db2 supports sophisticated statistics to help the optimizer compute better cardinality estimates
- Column group statistics help the optimizer recognize data correlation across a set of equality predicates
- Statistical views improve the optimizer's selectivity estimates for complex predicates and relationships across tables
- LIKE statistics help the optimizer's estimates for LIKE '%string' predicates

46

Tridex
A DB2 USER GROUP

IBM

Speaker: John Hornibrook
Company: IBM Canada
Email Address: jhornibr@ca.ibm.com

47

John is a Senior Technical Staff Member responsible for relational database query optimization on IBM's distributed platforms. This technology is part of Db2 for Linux, UNIX and Windows, Db2 Warehouse, Db2 on Cloud, IBM Integrated Analytics System (IIAS) and Db2 Big SQL. John also works closely with customers to help them maximize their benefits from IBM's relational DB technology products.