Insight
Americas 07

TALES FROM A UNIVERSE NINJA: PART TWO

David G. Rathbun, Integra Solutions

Business Objects™

BREAKOUT INFORMATION

- Tales from a Universe Ninja, Part Two
  - Training classes and product manuals can take you only so far. What do you do when the manual stops, but the project requirements do not? In this two-part session, find out how a team of experienced universe designers solved real-world universe challenges. Get tips, many of which are applicable to versions prior to BusinessObjects XI Release 2. Part two covers both Designer and Web Intelligence. See how a designer and report developer can work together to overcome challenges, such as dynamic report layout, measure swapping, and prompt handling. Download sample files after the session. Add to your arsenal of universe practices, and be at peace with the universe.

Print Information (please leave for Business Objects use)

Print Code

AGENDA

1. Introduction
2. Derived table tricks
3. List of values tricks
4. Conclusion
5. Q&A

INTRODUCTION

WHO IS DAVE?

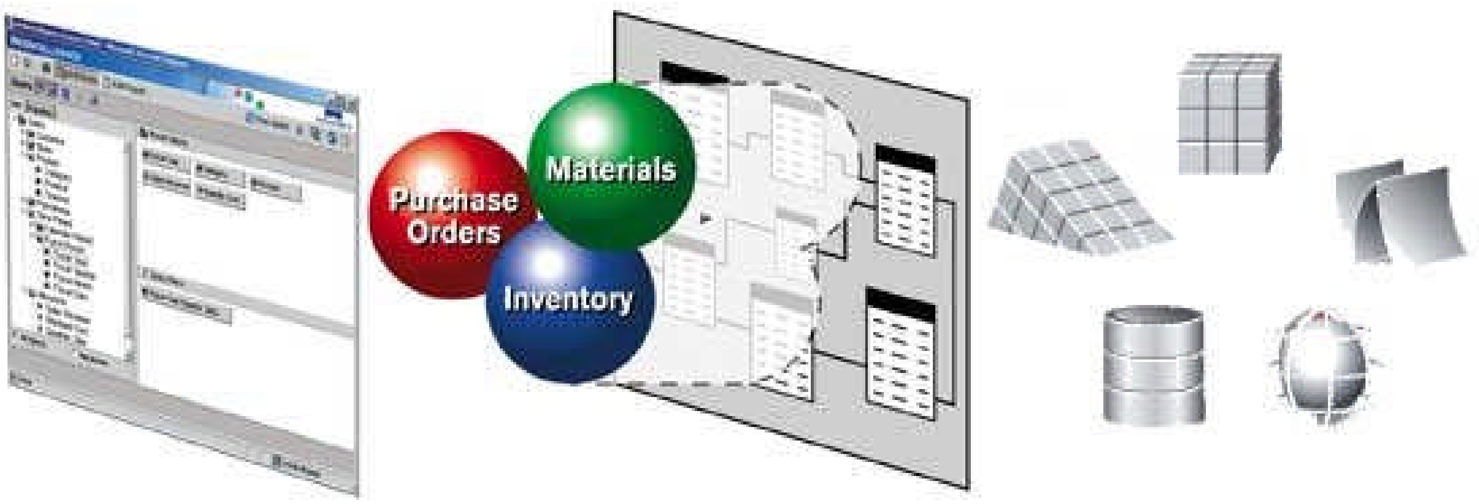
- Dedicated to BusinessObjects solutions since 1995
  - Consulting / mentoring / troubleshooting
  - Primary focus on knowledge transfer and client education
- Selected to present at 1996 - 2007 user conference
  - 12 consecutive years as a presenter
- Charter member of BOB
  - http://busobj.forumtopics.com

BOB
BusinessObjects
Board

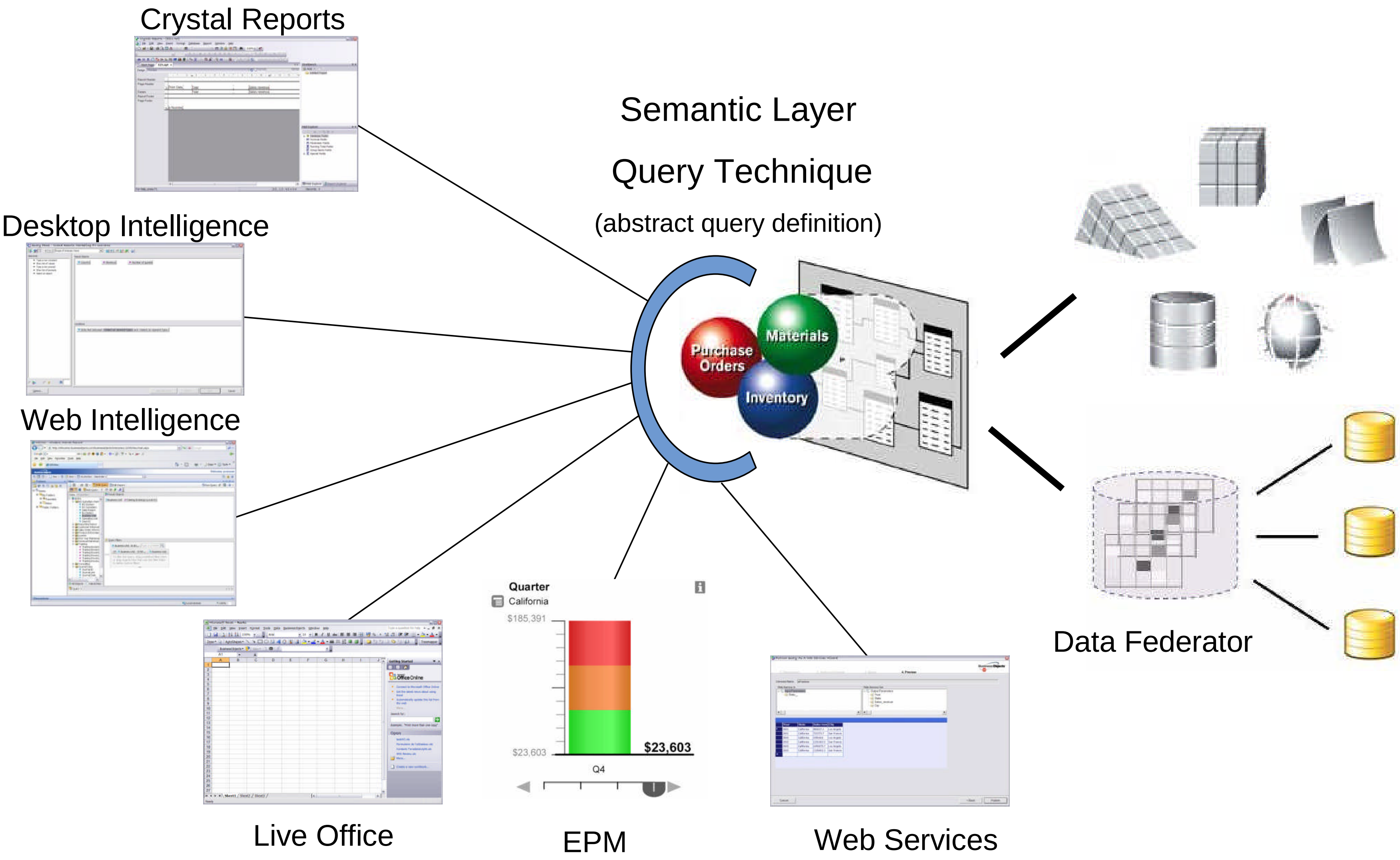
- I Blog! Dave’s Adventures in Business Objects
  - http://www.dagira.com

WHAT IS UNIVERSE DESIGNER?

- Provides a business representation of corporate data for end users
- They can access data autonomously using their everyday vocabulary
- Allows IT to keep control of the data access with fine-grained security



WHERE CAN YOU USE UNVIVERSE DESIGNER?



Crystal Reports

Desktop Intelligence

Web Intelligence

Semantic Layer

Query Technique

(abstract query definition)

Live Office

EPM

Web Services

Data Federator

DEMONSTRATION NOTES

- Demonstration universes
  - Island Resorts Marketing
  - XTreme sample universe
- Island Resorts was converted to Oracle
  - Required in order to demonstrate advanced features
  - Full outer joins
  - JOIN_BY_SQL
- Full software configuration
  - BusinessObjects Enterprise XI Release 2 Service Pack 2
  - Microsoft SQL Server is used for the repository
  - Oracle 10g is used for the Island Resorts databases

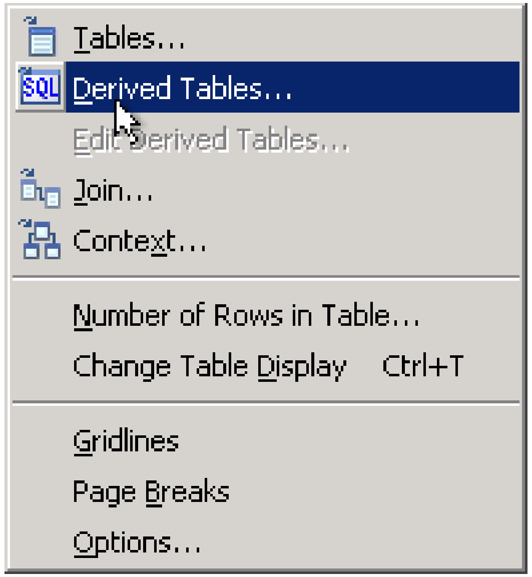
DERIVED TABLE TRICKS

Opportunities multiply as they are seized.

– Sun Tzu

DERIVED TABLES

- In most cases your content exists in a data source
- Derived tables were introduced in Universe Designer 6.5
  - Can include @Prompt() or @Variable function calls
  - Cannot include @Select(), @Where(), or @Aggregate_Aware()
- Some typical uses of this feature include
  - Substitute for views within the universe
  - Including prompts in views
  - Joining tables without using Universe Designer
- Creating a derived table is simple
  - Insert + derived table from the menu
  - Right-click and select derived table



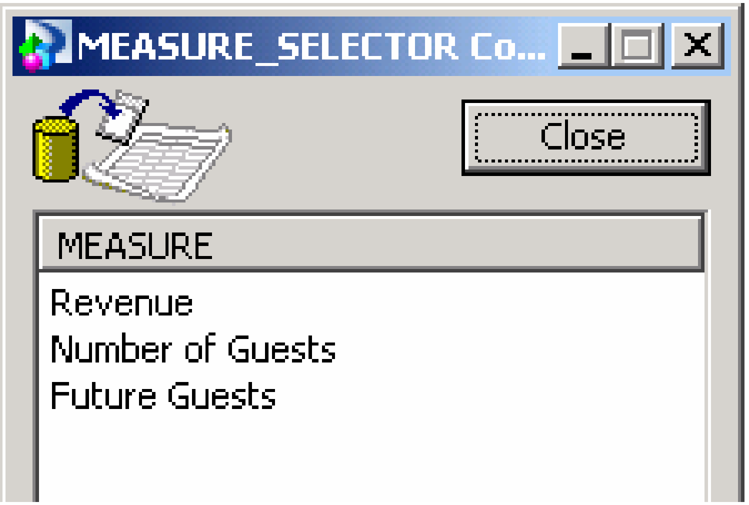
CREATING FAKE DATA

- Requirement: Allow a user to dynamically select a measure on a report
- Example: Island Resorts Marketing
  - Sales revenue
  - Number of guests
  - Future guests
- Solution 1: Prompt for a measure use the result in a formula (not shown today)
- Solution 2: Use “fake” data and drill filters

BUILD THE DERIVED TABLE

- Derived table for measure selector

```
select 'Revenue' as MEASURE from dual
UNION
select 'Future Guests' from dual
UNION
select 'Number of Guests' from dual
```

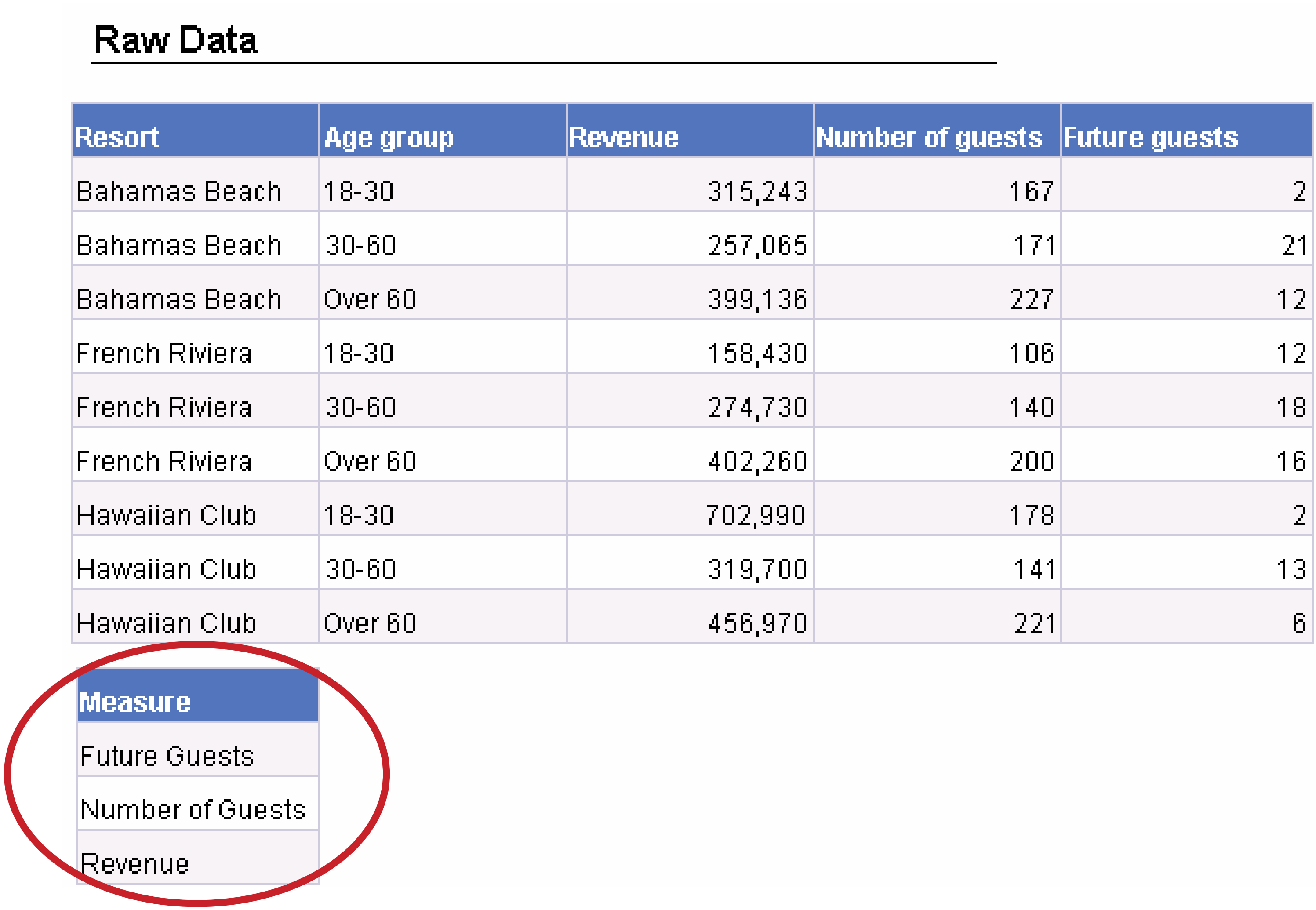


MEASURE
Revenue
Number of Guests
Future Guests

💻 Demonstration 1 – Building a derived table to create fake data for drill filter

INCLUDE THE FAKE DATA

- Query the fake data in a separate data provider



Raw Data

Resort	Age group	Revenue	Number of guests	Future guests
Bahamas Beach	18-30	315,243	167	2
Bahamas Beach	30-60	257,065	171	21
Bahamas Beach	Over 60	399,136	227	12
French Riviera	18-30	158,430	106	12
French Riviera	30-60	274,730	140	18
French Riviera	Over 60	402,260	200	16
Hawaiian Club	18-30	702,990	178	2
Hawaiian Club	30-60	319,700	141	13
Hawaiian Club	Over 60	456,970	221	6

Measure
Future Guests
Number of Guests
Revenue

REFERENCE THE DRILL FILTER

- This variable named analytic references the results of the DrillFilters() function

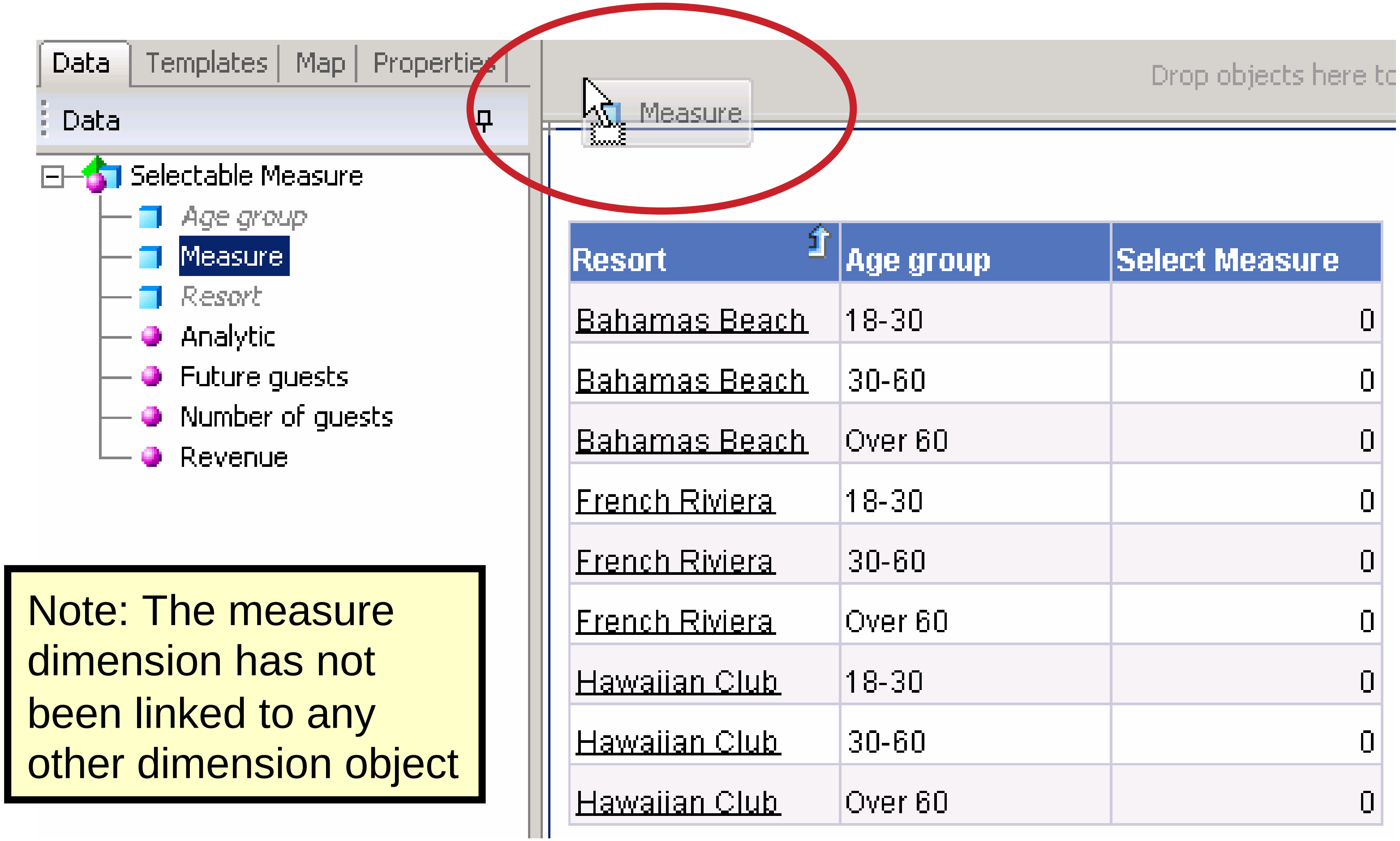
```
=If(DrillFilters([Measure])="Revenue";[Revenue];
If(DrillFilters([Measure])="Future Guests";[Future
  guests];
If(DrillFilters([Measure])="Number of
  Guests";[Number of guests];0)))
```

- This formula is used in the header of the column

```
=If(DrillFilters([Measure])="";"Select Measure";
DrillFilters([Measure]))
```

SET UP THE DRILL FILTER

- Drag the derived table object to the drill filters toolbar



Resort	Age group	Select Measure
Bahamas Beach	18-30	0
Bahamas Beach	30-60	0
Bahamas Beach	Over 60	0
French Riviera	18-30	0
French Riviera	30-60	0
French Riviera	Over 60	0
Hawaiian Club	18-30	0
Hawaiian Club	30-60	0
Hawaiian Club	Over 60	0

Note: The measure dimension has not been linked to any other dimension object

💻 Demonstration 2 – Dynamic report content with a fake data

DRILL FILTERS FOR DYNAMIC CONTENT

- Using a drill filter allows a report to pivot data
  - Revenue and number of guests are columns
  - Drill filter technique allows swapping of measures on a row
- @Prompt() could have been used
  - Requires a report refresh to swap measures
- A derived table with a union could have been used
  - Triples the size of the result set (three measures in a union)

```
SELECT resort, revenue, ‘Revenue’ as measure
Union
Select resort, number_of_guests, ‘Number of Guests’
Union
Select resort, future_guests, ‘Future Guests’
```

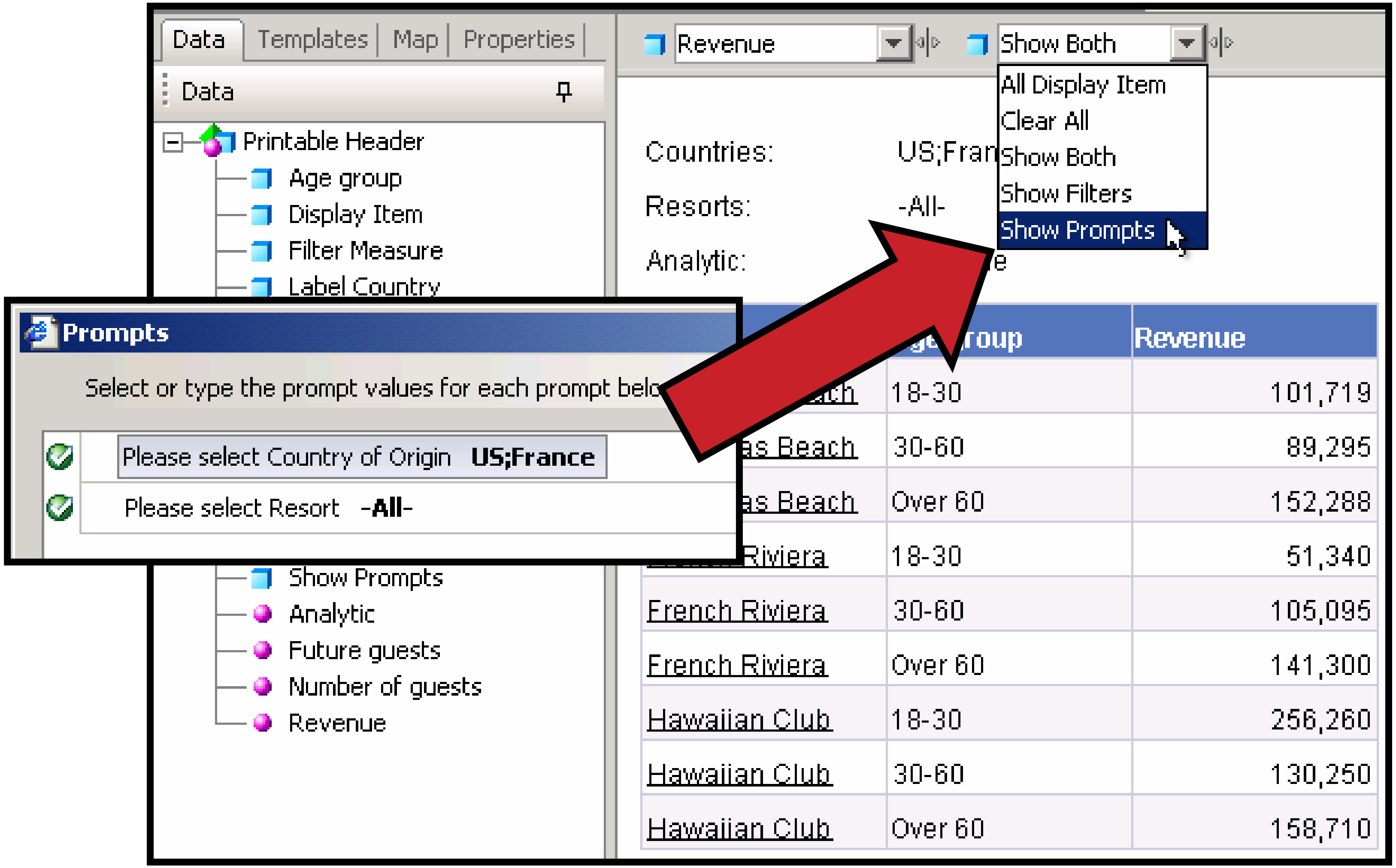
MORE FAKE DATA: PRINT ELEMENTS

- Build a derived table to include display options

```
select 'Clear All' as DISPLAY_ITEM from dual
UNION
select 'Show Both' as DISPLAY_ITEM from dual
UNION
select 'Show Filters' as DISPLAY_ITEM from dual
UNION
select 'Show Prompts' as DISPLAY_ITEM from dual
```

- Include the Display Item object in a second data provider
- Create the appropriate variables
- Display these in an “empty” table at the top of the report
- Allows user to preserve drill filters or prompt selections when printing

SELECTABLE PRINT ITEMS



Select or type the prompt values for each prompt belo

Please select Country of Origin **US;France**

Please select Resort **-All-**

Countries: US;Fran

Resorts: -All-

Analytic: e

	roup	Revenue
ach	18-30	101,719
as Beach	30-60	89,295
as Beach	Over 60	152,288
Riviera	18-30	51,340
French Riviera	30-60	105,095
French Riviera	Over 60	141,300
Hawaiian Club	18-30	256,260
Hawaiian Club	30-60	130,250
Hawaiian Club	Over 60	158,710

💻 Demonstration 3 – Display dynamic printable items

LIST OF VALUES TRICKS

He who knows when he can fight and when he cannot, will be victorious.

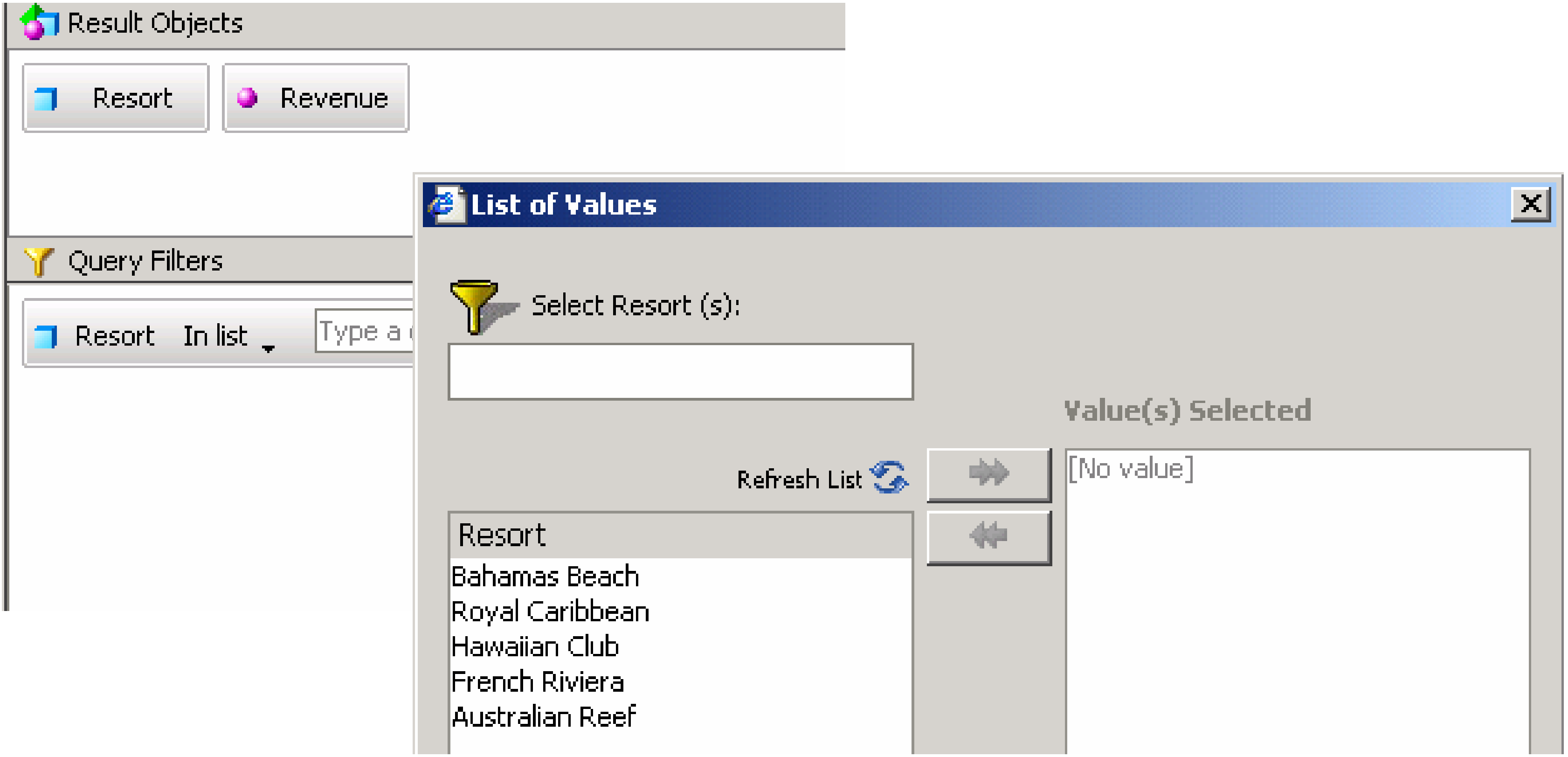
– Sun Tzu

AGENDA

1. What is a List of Values?
2. Using “All” in a List of Values
3. Cascading Lists of Values
4. Using “All” in a Cascading List of Values

WHAT IS A LIST OF VALUES?

- A list of values (LOV) query is used for convenience
- Provides a pick list of condition (filter) values



LOV QUERIES FOR PROMPTS

- Universe designers often customize a list of values query to support a prompt
  - Very convenient ☺
- When a prompt is added to a report it must be filled
  - Not so convenient ☹
- Canned reports lose some flexibility when every prompt must include at least one value
- What can be done to avoid this penalty?

AGENDA

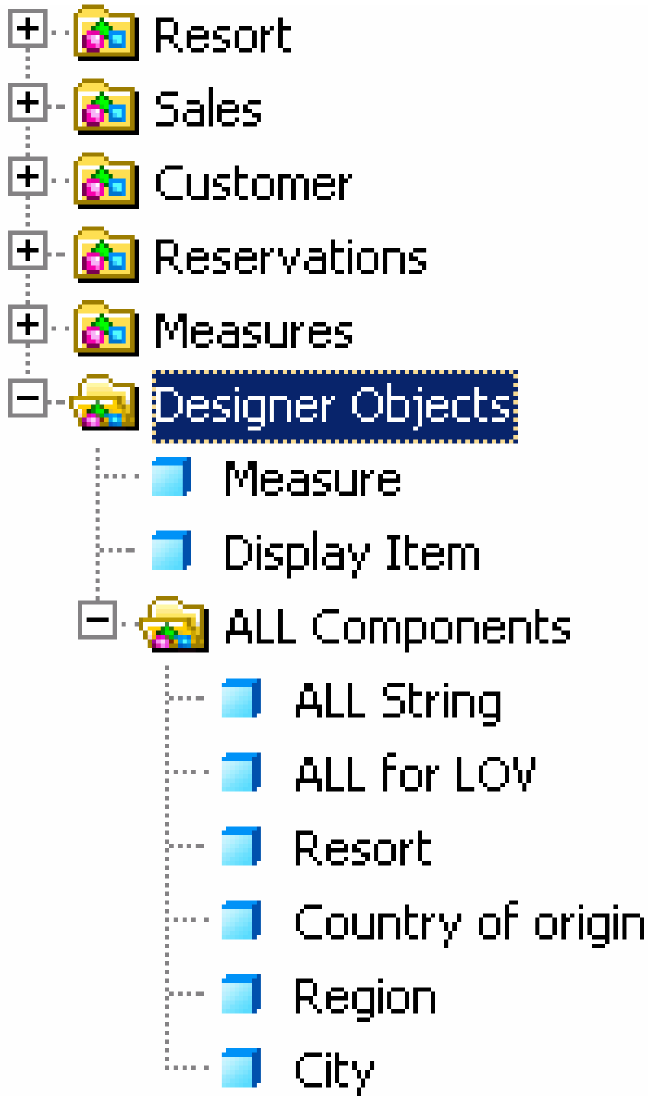
1. What is a List of Values?
2. Using “All” in a List of Values
3. Cascading Lists of Values
4. Using “All” in a Cascading List of Values

“ALL” SIMULATES OPTIONAL PROMPTS

- The “ALL” technique has been standardized and used for many years
  - Add “ALL” to a list of values
  - Include logic to process ALL within the predefined condition
- Requires a custom LOV
- Can have a performance impact
- Logic cannot be replicated on the query panel without special directions
- Does not work with index aware objects

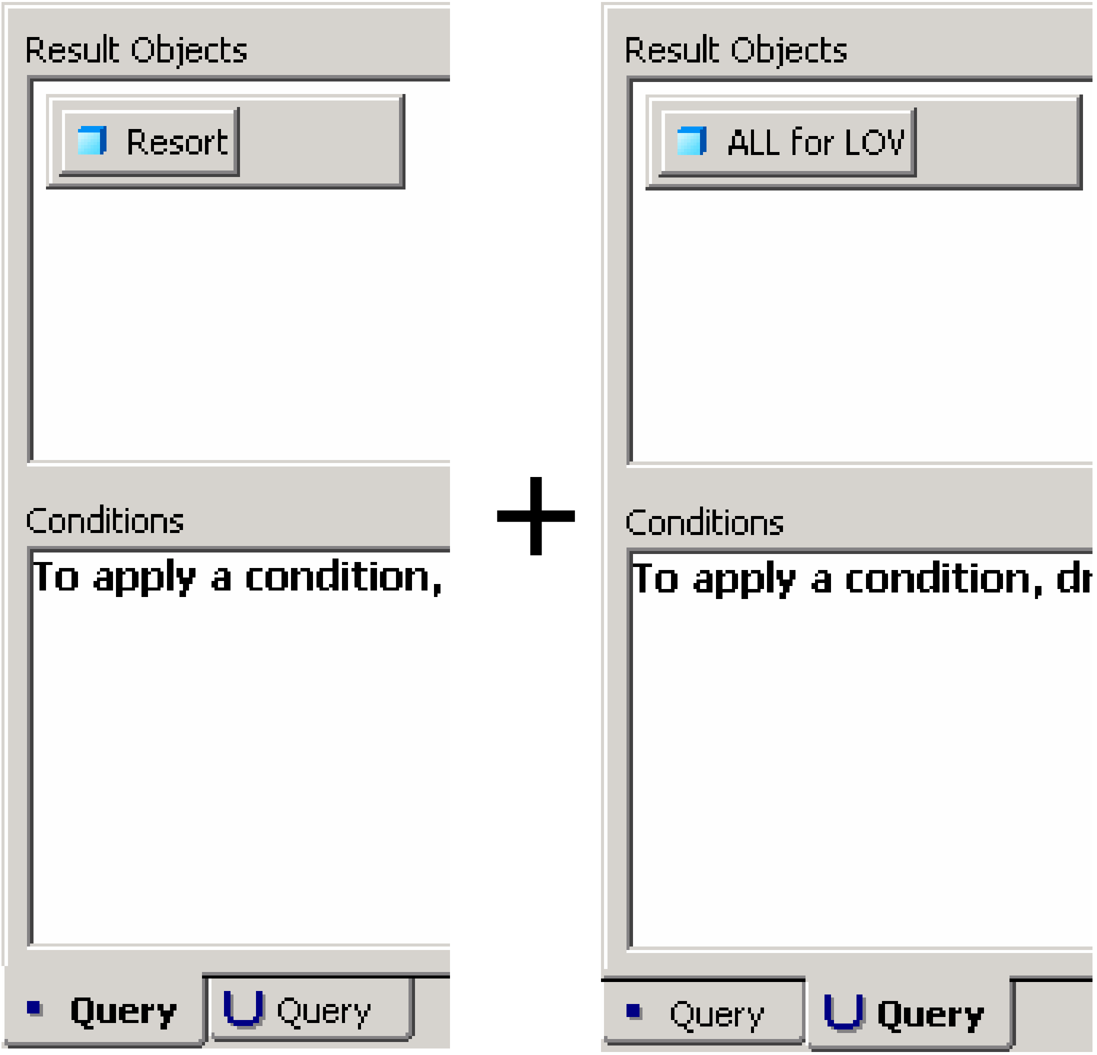
TRIED AND TRUE SOLUTION

- Create a hidden Universe Designer class
- Two “ALL” objects
    - String constant ‘-All-’
    - String constant that references a table
    - One used in prompts, other in LOV
- The format -All- ensures that the token will appear before all other values for ease of use
- Objects are copied so that the standard objects do not include “ALL” in the list
    - Otherwise users may pick it when it does not work



“ALL” LOV QUERY

```
SELECT DISTINCT
  RESORT.resort
FROM RESORT
UNION
SELECT ‘-ALL-’
FROM DUAL;
```



Note: The syntax shown is for an Oracle database. Syntax for other systems will be different.

“ALL” CONDITION OBJECT CODE

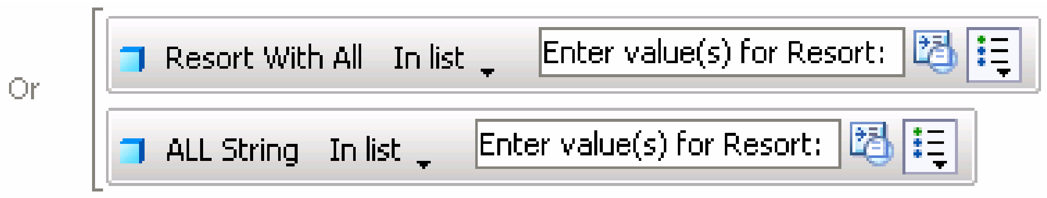
- Resort In (Selection) OR ALL in (Selection)
- When a resort is selected
  - Bahamas Beach In Bahamas Beach OR ALL In Bahamas Beach
  - The first half of the expression is true
- When ALL is selected
  - ALL in Resort List OR ALL in ALL
  - The second half of the expression is true for every possible resort

```
( @Select(Resort\Resort) IN @Prompt('Please select
Resort','A','ALL
Components\Resort',multi,constrained) OR
@Select(ALL Components\ALL String) IN
@Prompt('Please select Resort','A','ALL
Components\Resort',multi,constrained) )
```

💻 Demonstration 4 – Standard ALL Prompt Logic

“ALL” CONDITION ON THE QUERY PANEL

- Users can also create this logic, but at a cost
  - The object must include ALL in the LOV query
  - The ALL object must be public
  - The user must create a combined condition



- It is much easier to create this logic for the user in the universe
- Make sure that you do not include ALL in a LOV for any standard object to avoid confusion
  - Without the proper logic a user will get zero matching rows

💻 Demonstration 5 – Query panel ALL prompt logic

ISSUES WITH “ALL”

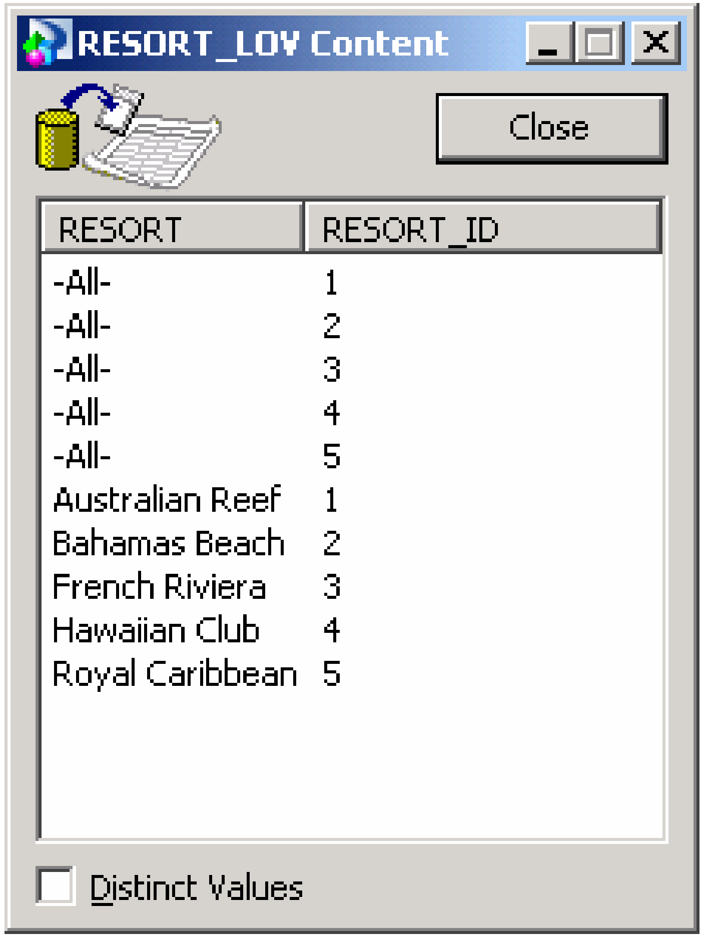
- Predefined conditions are generally required to avoid confusion
- Performance can be a problem
  - Teradata queries take two or more minutes with this logic and 10 seconds without
  - Does not remove tables from the query if they are not required
- Does not work with index awareness
  - Index aware adds extra objects to the LOV query that are not compatible with this technique
- Does not work with cascading list of values

AN OPTION FOR PERFORMANCE

- The “OR” logic is what trips up most database optimizers
- By creating a derived table we can eliminate the OR

```
SELECT RESORT.RESORT,
RESORT.RESORT_ID
FROM RESORT
UNION
SELECT '-All-',
RESORT.RESORT_ID
FROM RESORT
```

- The constant “ALL” is combined with every possible value

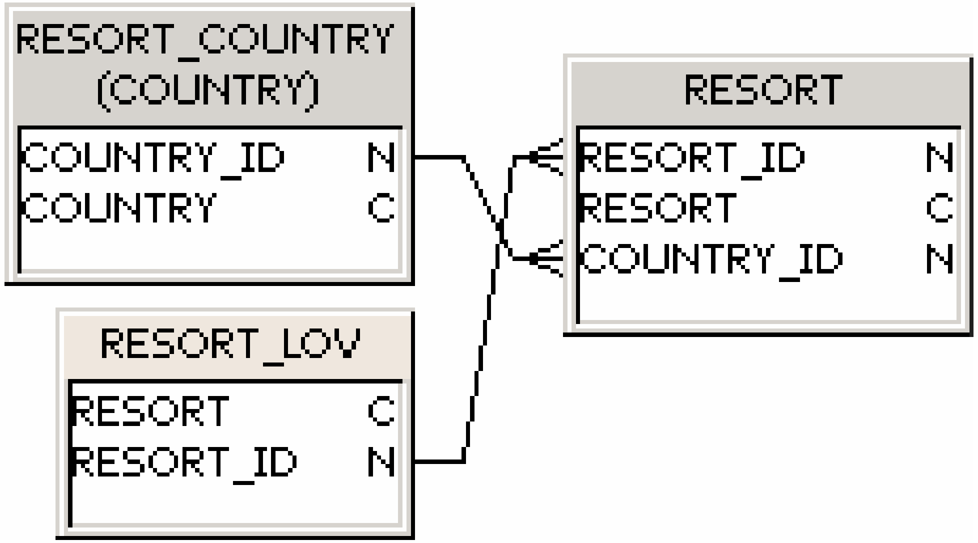


RESORT	RESORT_ID
-All-	1
-All-	2
-All-	3
-All-	4
-All-	5
Australian Reef	1
Bahamas Beach	2
French Riviera	3
Hawaiian Club	4
Royal Caribbean	5

☐ Distinct Values

AN OPTION FOR PERFORMANCE

- Join this new derived table to the existing dimension table using a unique key
- Set the cardinality to one to many



- Oracle case study: reduced query times from minutes to seconds

IMPROVED “ALL” CONDITION OBJECT

- The new logic no longer requires an OR condition

```
RESORT_LOV.RESORT IN @Prompt('Please select
Resort','A','LOV with Derived Table
Join\Resort',multi,constrained)
```

- Even with the additional join the performance can improve because of the query logic
- Your mileage may vary
  - Oracle seems to work fine
  - Teradata is not fooled by this trick
- Consider replacing derived table with a physical table with indexes

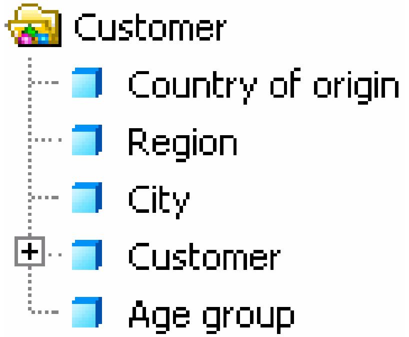
💻 Demonstration 6 – Derived table join ALL prompt logic

AGENDA

1. What is a List of Values?
2. Using “All” in a List of Values
3. Cascading Lists of Values
4. Using “All” in a Cascading List of Values

CASCADING LIST OF VALUES

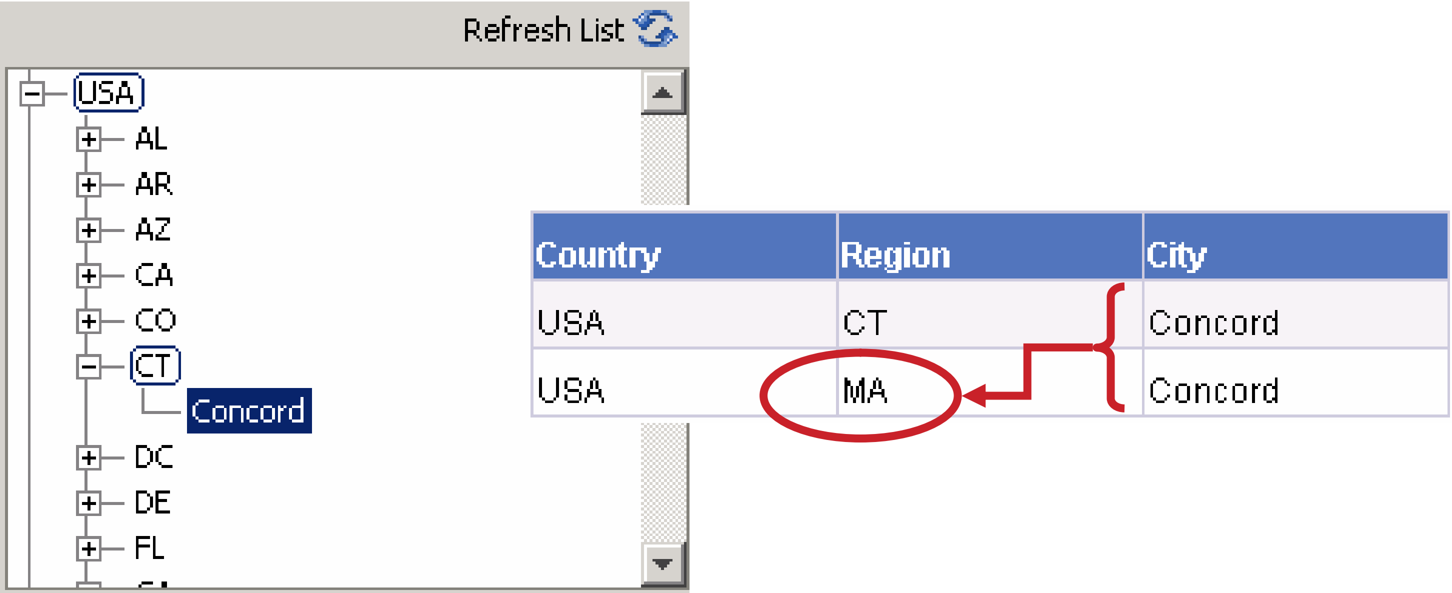
- A hierarchy can be used to filter a list of values
  - Pick a country
    - Pick a region
      - Pick a city
      - Run the query
- Universe Designer offers a built-in cascading option
- This feature is essentially a shortcut for a technique that experienced designers had been using for years



💻 Demonstration 7 – Building a cascading LOV in Universe Designer

DRAWBACKS OF CASCADING LOV QUERIES

- A user cannot stop in the middle of a selection
- Only the final selection is applied to the query logic
- The standard ALL technique does not cascade
- Sample query from the Xtreme sample universe



Country	Region	City
USA	CT	Concord
USA	MA	Concord

💻 Demonstration 8 – Cascading LOV with non-unique final selection

AGENDA

1. What is a List of Values?
2. Using “All” in a List of Values
3. Cascading Lists of Values
4. Using “All” in a Cascading List of Values

“ALL” WITH CASCADING LOV QUERIES

- With ~~a little~~ a lot of work a Universe Designer can create cascading LOV queries that…
  - Include ALL
  - Work with non-unique final selection value
- There does not appear to be a workaround for stopping in the middle of a cascading selection
  - Users must proceed all the way to the end of the cascade
- Including ALL in a cascading LOV requires
  - Custom objects for use in cascading queries
  - Concatenations of ALL + each level of the cascade
  - Combined UNION queries of increasing complexity at each level
  - Special condition object that processes the user selection

CUSTOM OBJECTS FOR CASCADING “ALL”

- Delimiter object

  ```
  ' ' || chr(187) || ' '
  ```

- All object

  ```
  '-All-'
  ```

- All object for LOV (references DUAL table)

  ```
  '-All-'
  ```

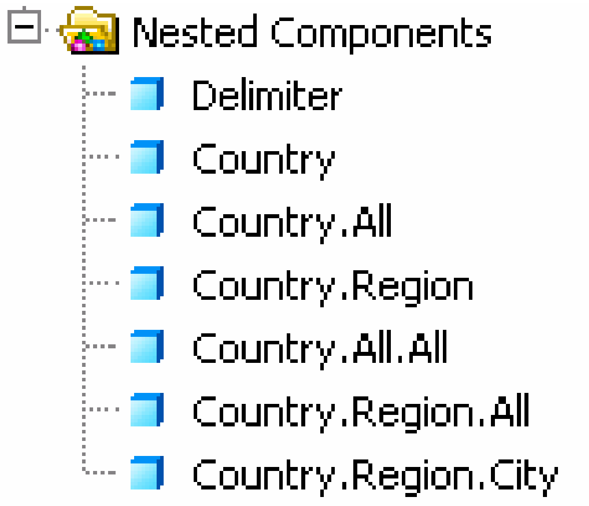
- Concatenated Objects

  ```
  REGION_COUNTRY.country || @Select(Nested
  Components\Delimiter) || @Select(ALL Components\ALL
  String)
  ```

CONCATENATED OBJECTS

- Country.All code

```
REGION_COUNTRY.country ||
@Select(Nested
Components\Delimiter) ||
@Select(ALL Components\ALL String)
```



- After substitution by @Select()

```
REGION_COUNTRY.country || ' ' ||
chr(187) || ' ' ||  '-All-'
```

- @Select() is for easy maintenance
  - Either “ALL” or delimiter can be changed with one edit

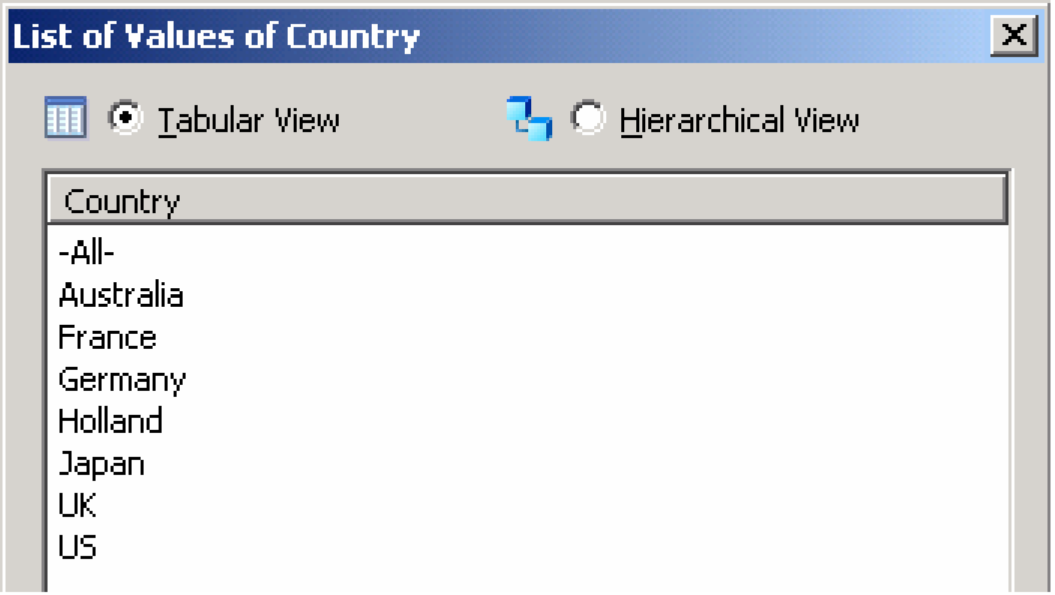
CASCADING “ALL” SCENARIO

- The goal is to create a three-step prompt that cascades
  - Country
  - Region
  - City
- The final selection must be unique
- User may see “ALL” at any level
- The user may select “ALL” only at the region or city level
  - I found that All – All – All selections generated SQL errors in Web Intelligence even though it worked in Desktop Intelligence
  - By forcing a selection of a real value at the top level the exception was avoided

CASCADING “ALL” STEP ONE

- Country LOV does include ALL

```
SELECT DISTINCT
  REGION_COUNTRY.country
FROM
  COUNTRY  REGION_COUNTRY
UNION
SELECT DISTINCT
  ( '-All-' )
FROM
  DUAL
```



Country
-All-
Australia
France
Germany
Holland
Japan
UK
US

CASCADING “ALL” STEP TWO PART ONE

- Object concatenation of Country.All

```
REGION_COUNTRY.country || ' ' || chr(187) || ' '
||  '-All-'
```

- Object concatenation of Country.Region

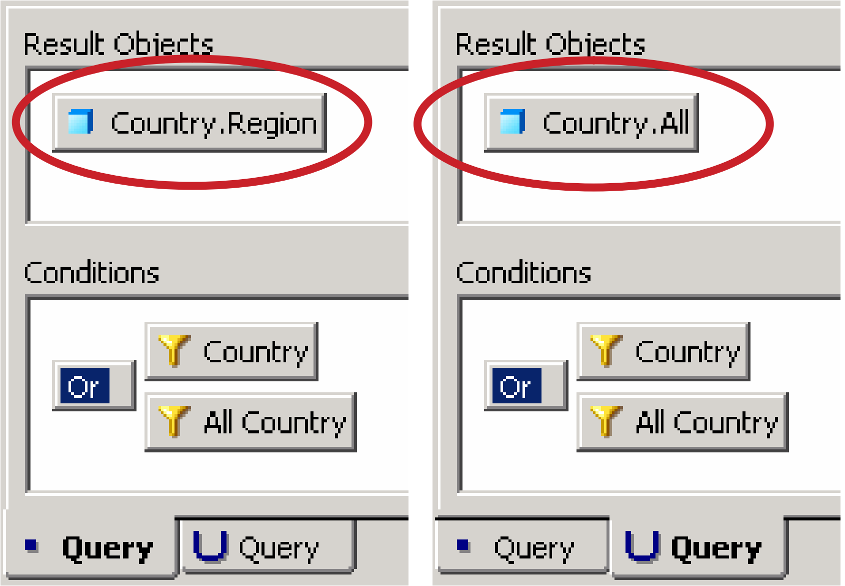
```
REGION_COUNTRY.country || ' ' || chr(187) || ' '
|| REGION.region
```

- Create condition objects that process the ALL

```
REGION_COUNTRY.country IN @Prompt('Please select
Country of Origin','A','Nested
Components\Country',multi,constrained) OR '-All-'
IN @Prompt('Please select Country of
Origin','A','Nested
Components\Country',multi,constrained)
```

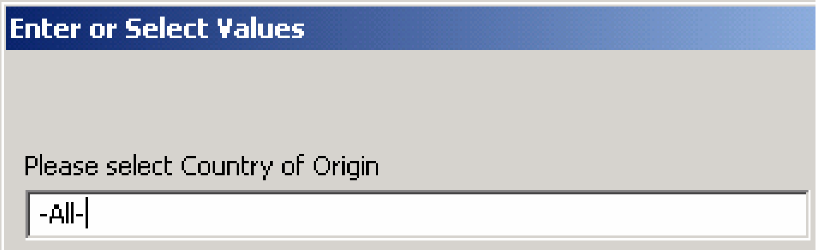
CASCADING “ALL” STEP TWO PART TWO

- Query panel for Country.Region list of values
- The condition objects below reference the country object with the “ALL’ that was created in step one
- Each side of the union allows for a specific country or ALL to be selected



CASCADING “ALL” STEP TWO RESULTS

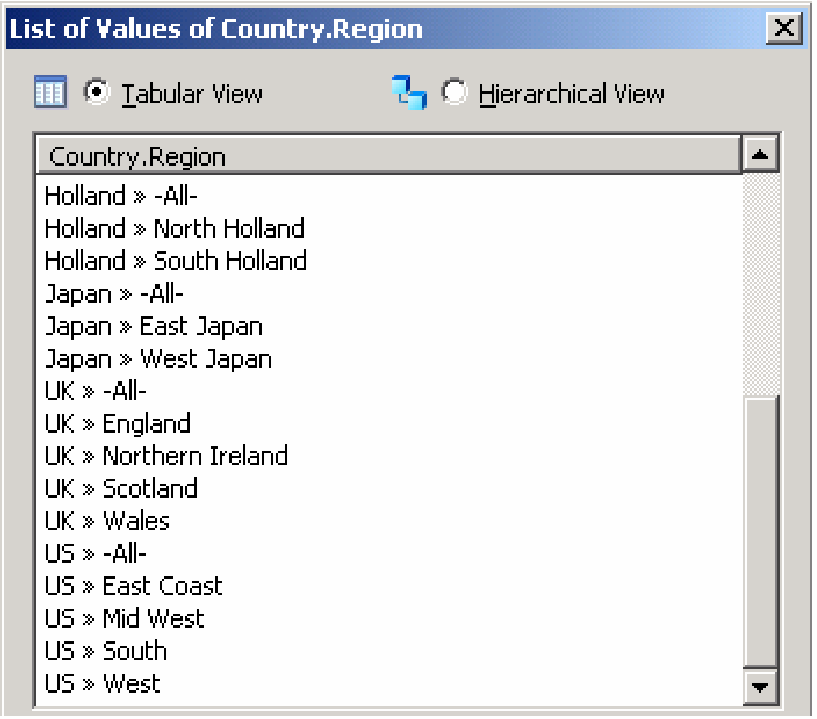
- Selected country = “ALL”



Enter or Select Values

Please select Country of Origin

-All-

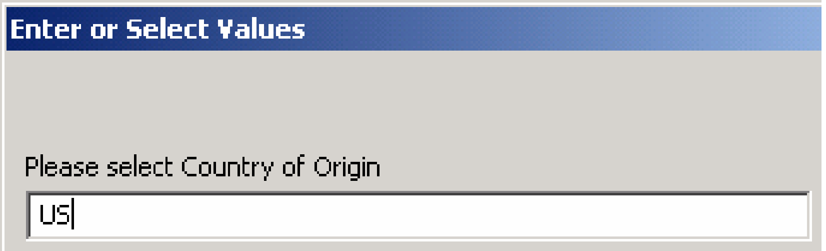


List of Values of Country.Region

● Tabular View ○ Hierarchical View

Country.Region
Holland » -All-
Holland » North Holland
Holland » South Holland
Japan » -All-
Japan » East Japan
Japan » West Japan
UK » -All-
UK » England
UK » Northern Ireland
UK » Scotland
UK » Wales
US » -All-
US » East Coast
US » Mid West
US » South
US » West

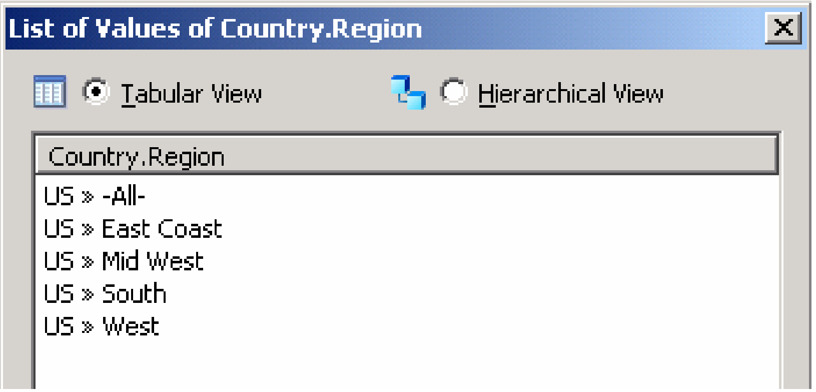
- Selected country = “US”



Enter or Select Values

Please select Country of Origin

US



List of Values of Country.Region

● Tabular View ○ Hierarchical View

Country.Region
US » -All-
US » East Coast
US » Mid West
US » South
US » West

As mentioned there is no All » All option on this list because of errors; the user must select a country.

CASCADING “ALL” STEP THREE PART ONE

- As before create concatenated objects
  - Country.All.All

```
REGION_COUNTRY.country || ' ' || chr(187) || ' ' ||
'-All-' || ' ' || chr(187) || ' ' || '-All-'
```

  - Country.Region.All

```
REGION_COUNTRY.country || ' ' || chr(187) || ' ' ||
REGION.region || ' ' || chr(187) || ' ' || '-All-'
```

  - Country.Region.City

```
REGION_COUNTRY.country || ' ' || chr(187) || ' ' ||
REGION.region || ' ' || chr(187) || ' ' || CITY.city
```

CASCADING “ALL” STEP THREE PART ONE

- And create condition objects
  - Country.Region

```
REGION_COUNTRY.country || ' ' || chr(187) || ' ' ||
'-All-' || ' ' || chr(187) || ' ' || '-All-'
```

  - Country.All

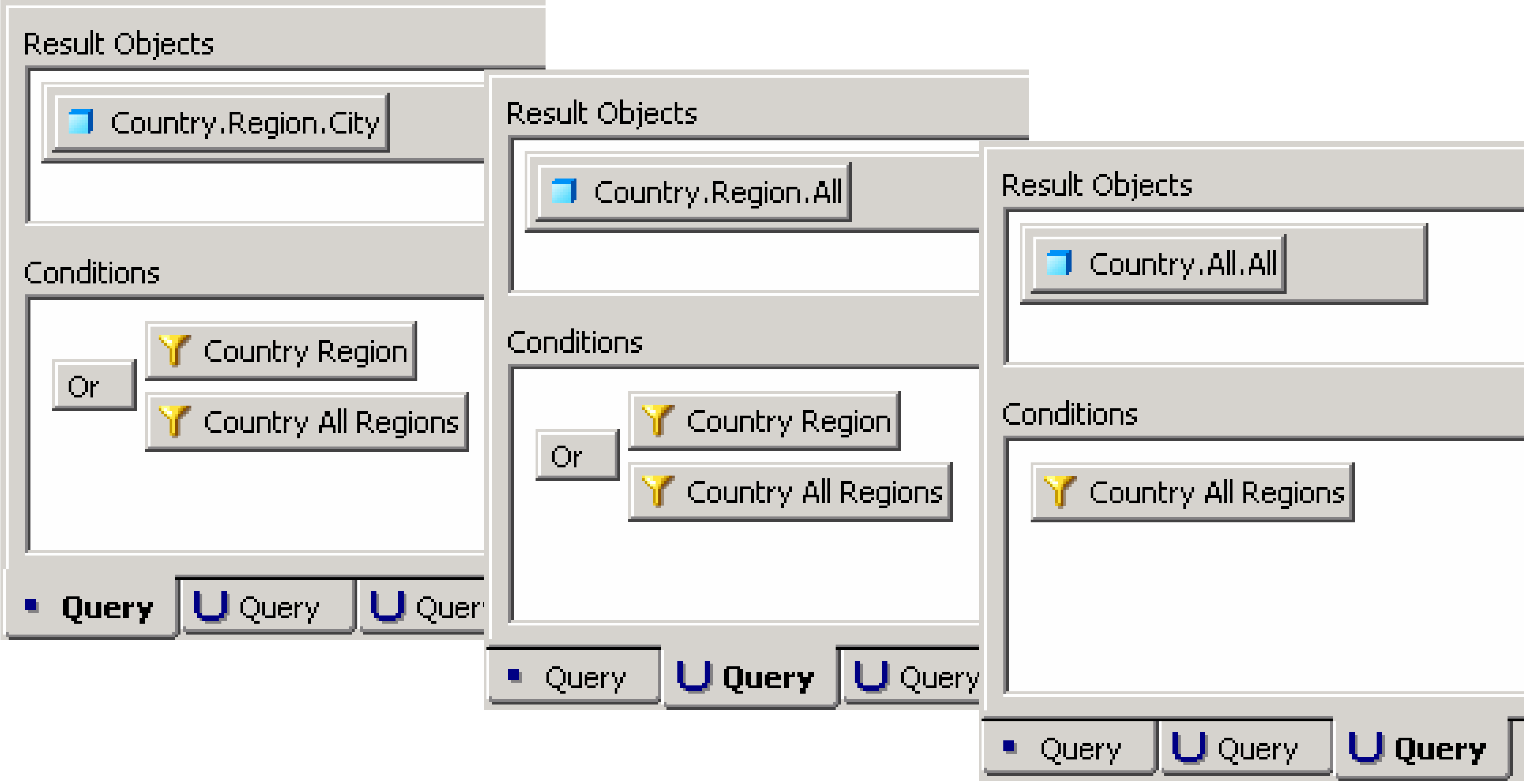
```
REGION_COUNTRY.country || ' ' || chr(187) || ' ' ||
REGION.region || ' ' || chr(187) || ' ' || '-All-'
```

  - All.All

```
REGION_COUNTRY.country || ' ' || chr(187) || ' ' ||
REGION.region || ' ' || chr(187) || ' ' || CITY.city
```

CASCADING "ALL" STEP THREE PART TWO

- Query panel for Country.Region.City list of values



Result Objects

Country.Region.City

Conditions

Or

Country Region

Country All Regions

Result Objects

Country.Region.All

Conditions

Or

Country Region

Country All Regions

Result Objects

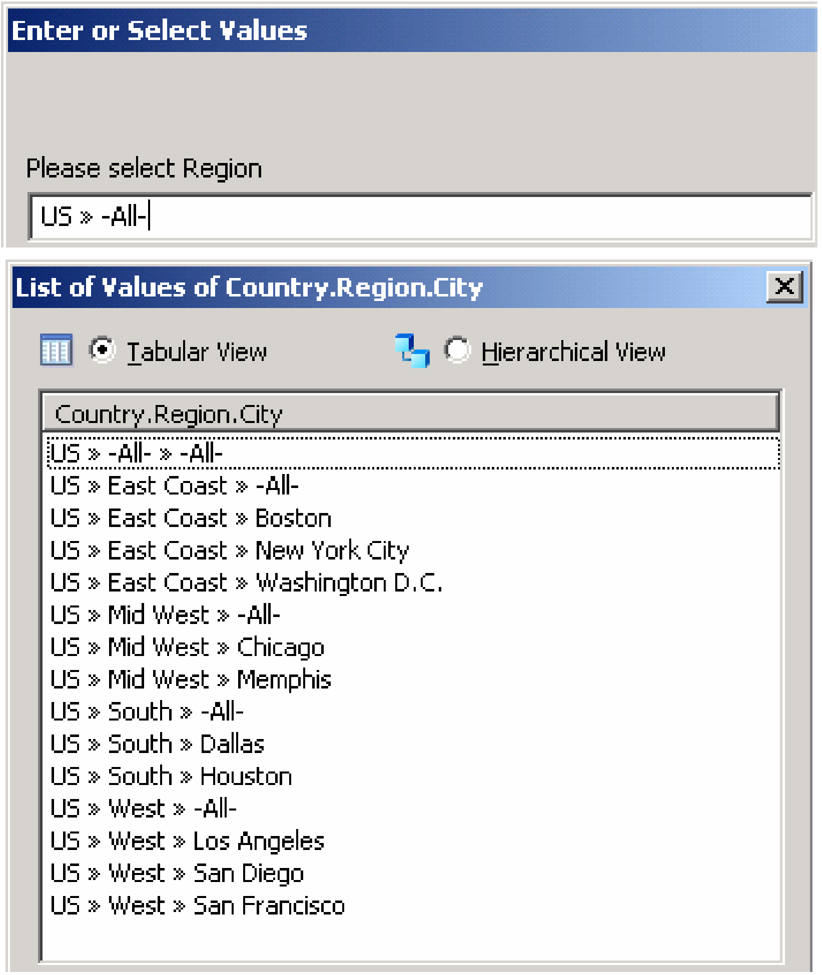
Country.All.All

Conditions

Country All Regions

CASCADING “ALL” STEP THREE RESULTS

- Selected Country = “US”
  Selected Region = “ALL”



Enter or Select Values

Please select Region

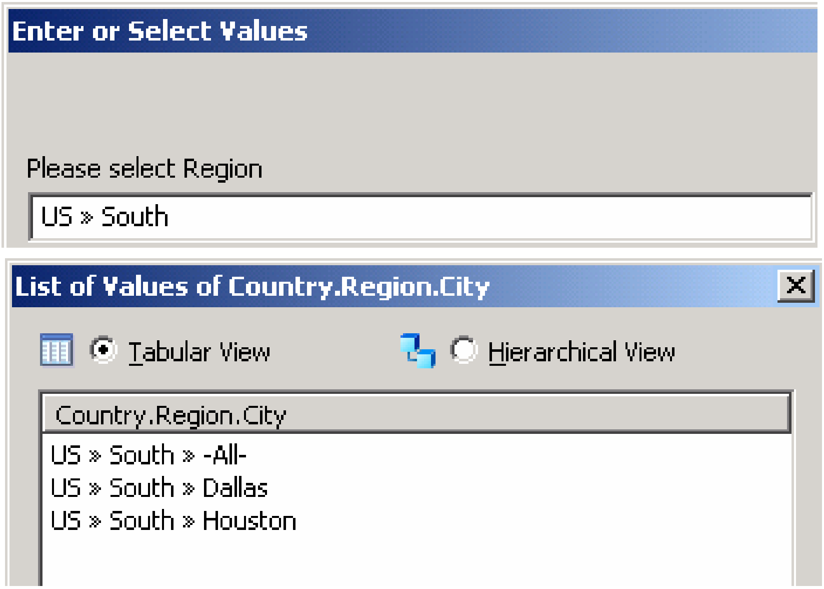
US » -All-

List of Values of Country.Region.City

Country.Region.City

US » -All- » -All-
US » East Coast » -All-
US » East Coast » Boston
US » East Coast » New York City
US » East Coast » Washington D.C.
US » Mid West » -All-
US » Mid West » Chicago
US » Mid West » Memphis
US » South » -All-
US » South » Dallas
US » South » Houston
US » West » -All-
US » West » Los Angeles
US » West » San Diego
US » West » San Francisco

- Selected Country = “US”
  Selected Region = “South”



Enter or Select Values

Please select Region

US » South

List of Values of Country.Region.City

Country.Region.City

US » South » -All-
US » South » Dallas
US » South » Houston

As mentioned there is no All » All option on this list because of errors; the user must select a country.

CASCADING “ALL” REVIEW

- For the last step, build a predefined condition that handles any / all of the possible selections
- Pseudo-code shown here

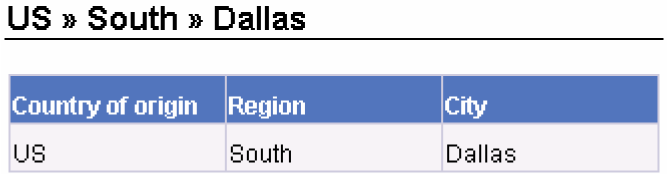
```
Country.Region.City IN @Prompt(…)
OR Country.Region.All IN @Prompt(…)
OR Country.All.All IN @Prompt(…)
OR All.All.All IN @Prompt(…)
```

- The last line is not shown today for reasons mentioned earlier
- The big question, does it work?

💻 Demonstration 9 – Cascading LOV with cascading ALL option

WHEW!

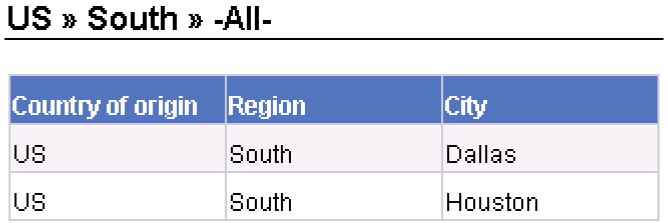
- Country + Region + City



US » South » Dallas

Country of origin	Region	City
US	South	Dallas

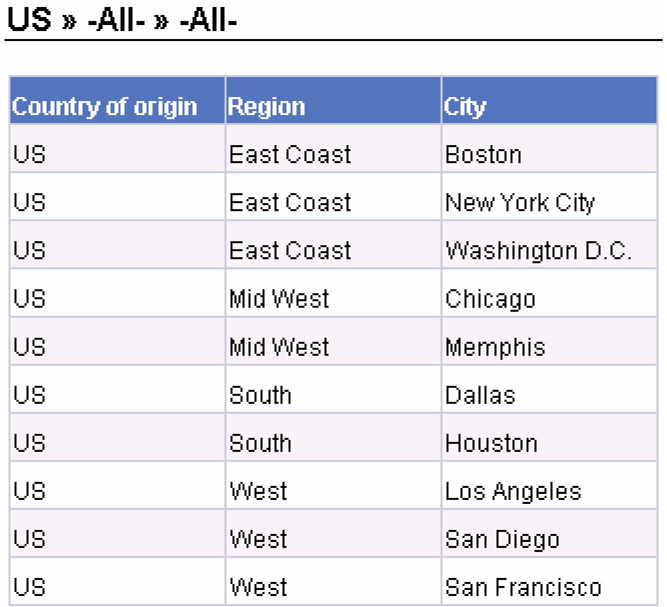
- Country + Region + All



US » South » -All-

Country of origin	Region	City
US	South	Dallas
US	South	Houston

- Country + All + All



US » -All- » -All-

Country of origin	Region	City
US	East Coast	Boston
US	East Coast	New York City
US	East Coast	Washington D.C.
US	Mid West	Chicago
US	Mid West	Memphis
US	South	Dallas
US	South	Houston
US	West	Los Angeles
US	West	San Diego
US	West	San Francisco

CONCLUSION

- Derived tables can be used for a variety of interesting purposes
  - Creating fake data
  - Optimizing ALL performance on LOV queries
- Prompts can benefit from LOV customizations
  - “ALL” logic can serve as optional prompts
  - Derived tables can be help with performance
  - Cascading prompts can also include “ALL”
- With some creativity – and desperation! – a universe designer can use standard features in unanticipated ways

Q&A

- Thank you for your time and attention today
- Questions
  - David G. Rathbun, Integra Solutions
- Contact information
  - 214 637 6622
  - www.IntegraSolutions.net

INTEGRA SOLUTIONS
a business unit of Quorum Business Solutions