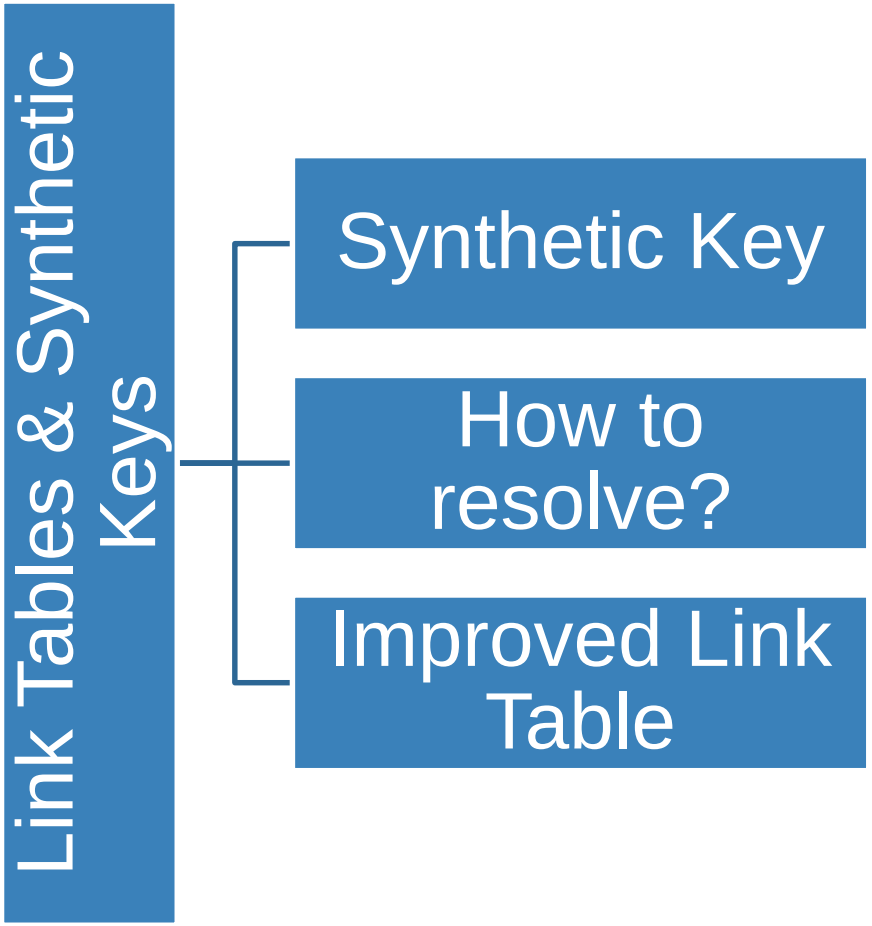
6. Link Tables and Synthetic Keys

VERTECHS.IO
TEACH..LEARN..INSPIRE

6.0: Chapter Goals



6.1: Introduction to Synthetic Keys and Link Tables

- QlikView associates two tables with a common field between them. Field name is case sensitive. This is called Qlik's Associative Join.

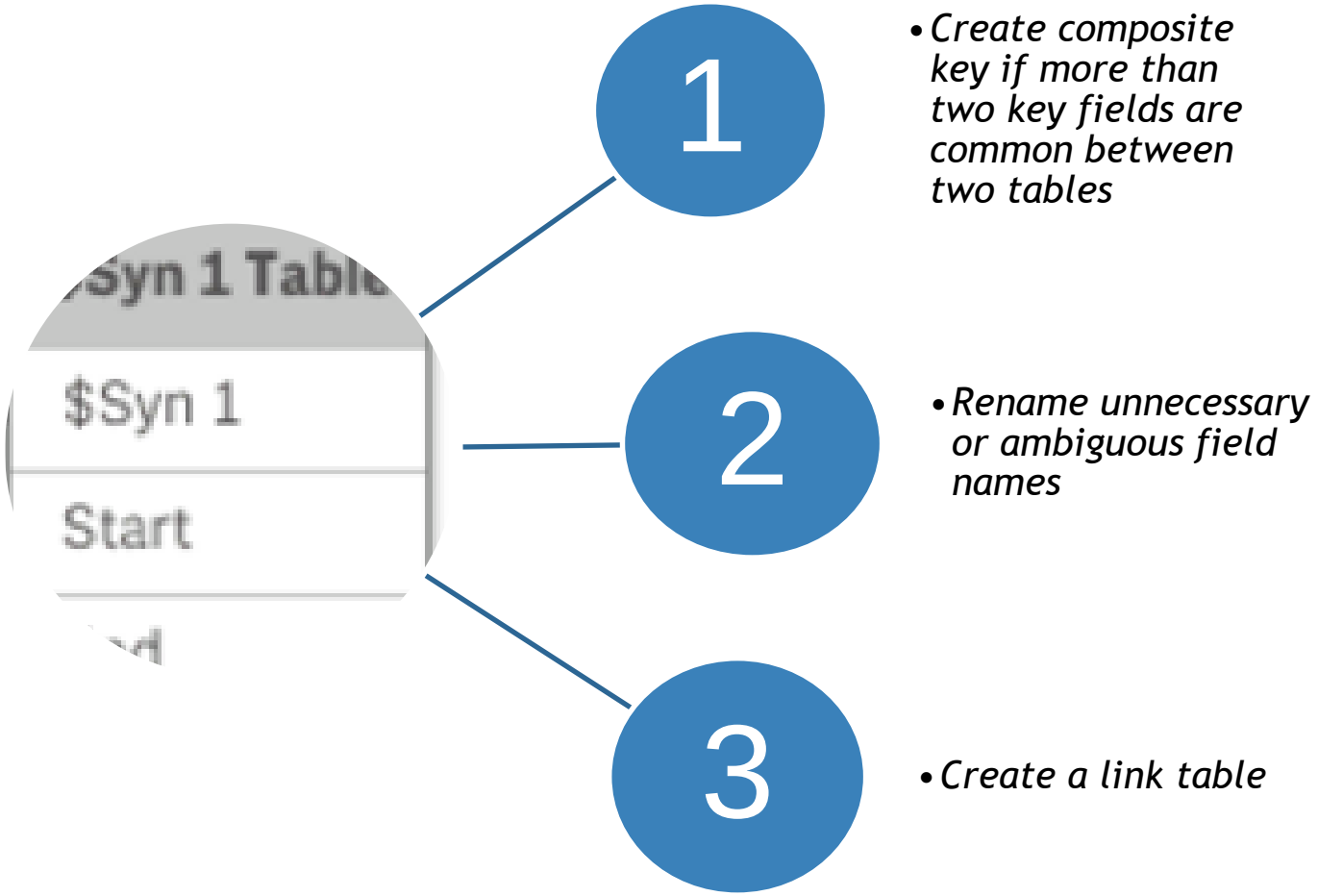
- When more than one common fields exist between two tables, QlikView creates a synthetic table and synthetic keys.

- Synthetic keys slow down performance

- A good data modeling practice is to resolve synthetic key.

- Three commonly used methods to resolve synthetic key

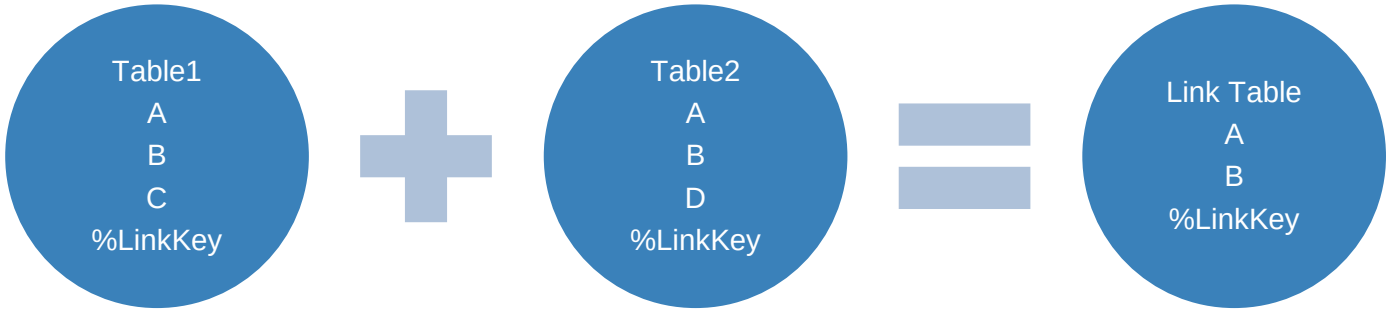
Ways to resolve synthetic keys



6.2: Link Table Deep Dive

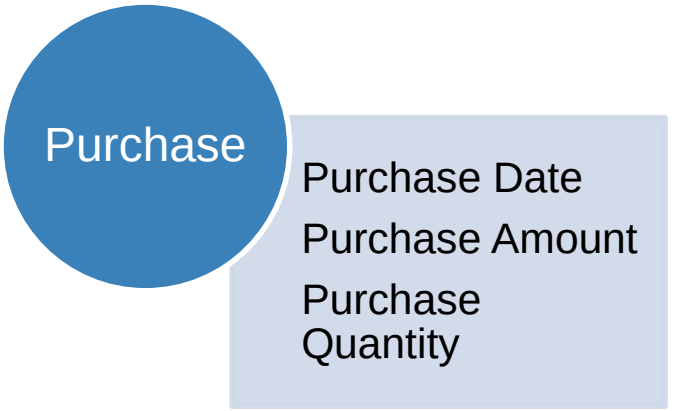
What is a link table?

- Link table is similar to a synthetic table
- Link table contains common fields between two tables



- Name Link Table key field using "%" as prefix along with “HidePrefix" variable assigned to "%".
- Only distinct combinations from both tables get stored in the link table
- Never aggregate or use Link Table key field as part of a measure. Because it will always be distinct values and it will show incorrect results.
- Two tables with different level of granularity: Sales and Budget
- Works well for smaller data sets but may not be best suitable for large fact tables

Challenge: Two Fact tables



Compare facts from both tables across shared dimensions

6.3: Creating Link Table - I

Steps to create a Link table

1. Add composite key in each table with more than one common fields
2. Make sure to prefix key field with % sign

Sales

%Key_OrderDate

%Key_ProductID

- %Key_OrderDate &'|'& %Key_ProductID as %LinkKey

Purchases

%Key_PurchaseDate

%Key_ProductID

- %Key_PurchaseDate &'|'& %Key_ProductID as %LinkKey

6.3: Creating Link Table - I

3. Load distinct records from the first table using resident load
4. Load distinct records from the second table using resident load
5. Concatenate both tables using concatenate operator.
6. Drop or rename common fields from both tables

Link Table

```
load
    Distinct
    %Key_OrderDate as %Key_Date,
    %Key_ProductID,
    %LinkKey
Resident Sales

;
```

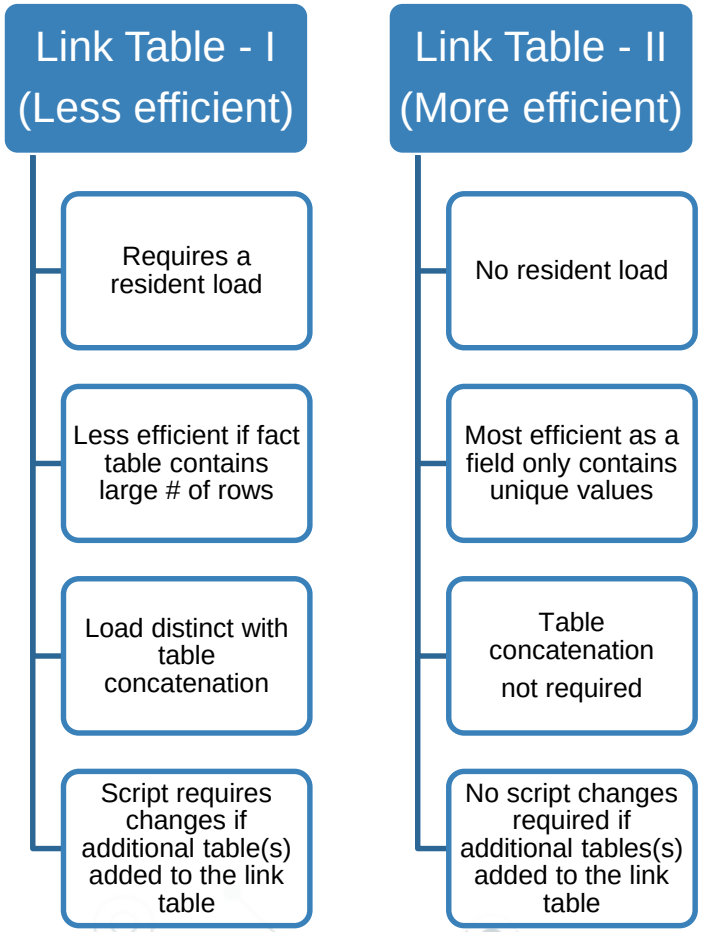
```
Concatenate([Link Table])

load
    Distinct
     %Key_PurchaseDate as %Key_Date,
     %Key_ProductID,
     %LinkKey
Resident Purchases
where not exists(%SalesLinkKey, %LinkKey)

;
```

```
drop fields %Key_OrderDate, %Key_PurchaseDate, %SalesLinkKey, %Key_ProductID from Sales, Purchases
```

6.4: Creating Link Table - II

Better way to create a Link table:

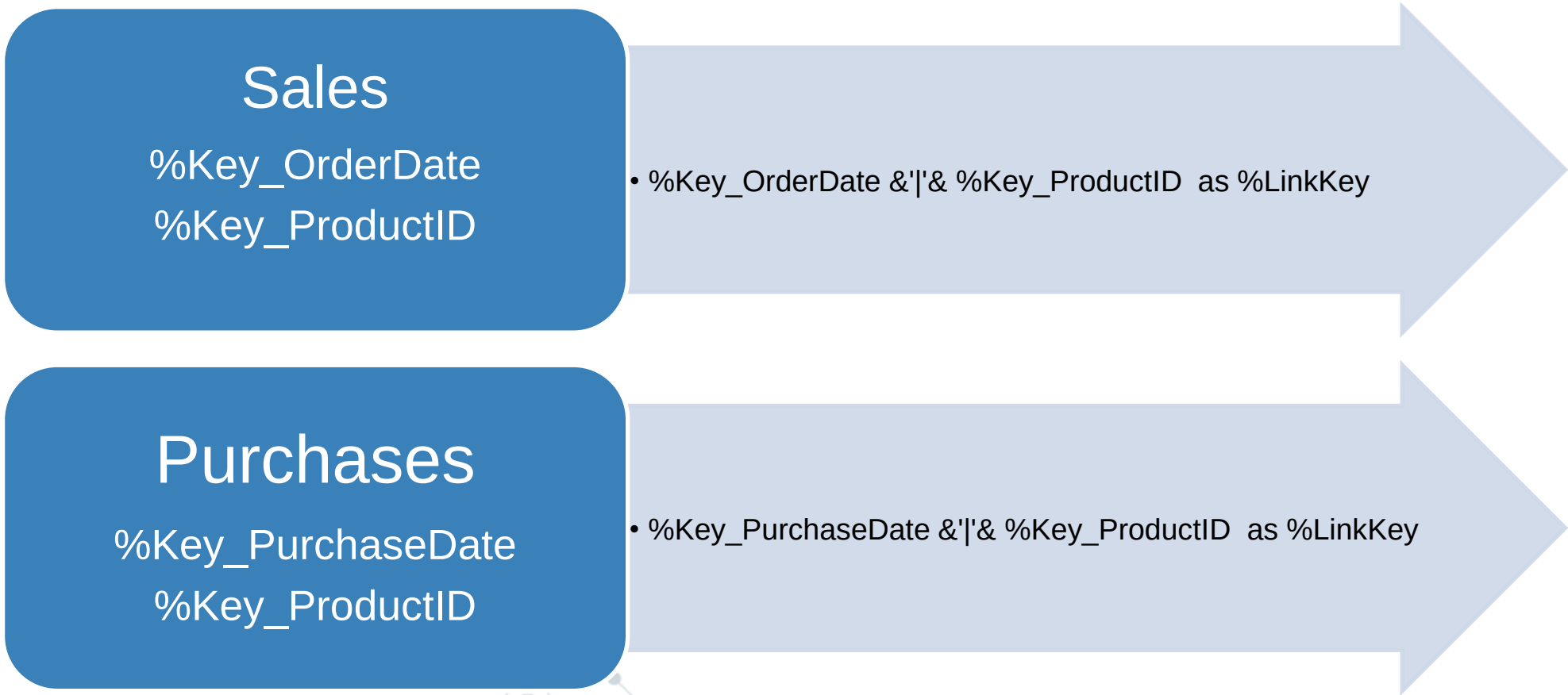
Use Fieldvalue() function instead of a resident load



6.4: Creating Link Table - II

Better way to create a Link table:

1. Add composite key in each table with more than one common fields



6.4: Creating Link Table - II

2. Create a link table using fieldvalue() function using the composite key
3. Perform preceding load to add each individual field using subfield() function
4. Drop common fields from both tables

Link Table

```
Link:
 load
      *,
      date(SubField(%LinkKey, '|',1), 'M/D/YYYY') as %Key_Date,
      SubField(%LinkKey, '|',2) as %Key_ProductID
;
```

```
LOAD
    FieldValue('%LinkKey', RecNo())                AS %LinkKey
AUTOGENERATE FieldValueCount('%LinkKey')
;
```

```
drop fields %Key_OrderDate, %Key_PurchaseDate, %SalesLinkKey, %Key_ProductID from Sales, Purchases
```

6.5: Challenge – Synthetic Keys to Link Table

Challenge

Create a link table to store Inventory Date and Product ID from the purchase table with Order Date and Product ID from the Sales table and Purchase Date and Product ID from the Purchase table into the link table.

Create the key field with % prefix and use HidePrefix variable to hide the link table key field.

Improve your link table by using Fieldvalue() function.

6.6: Solution – Synthetic Keys to Link Table

Solution