

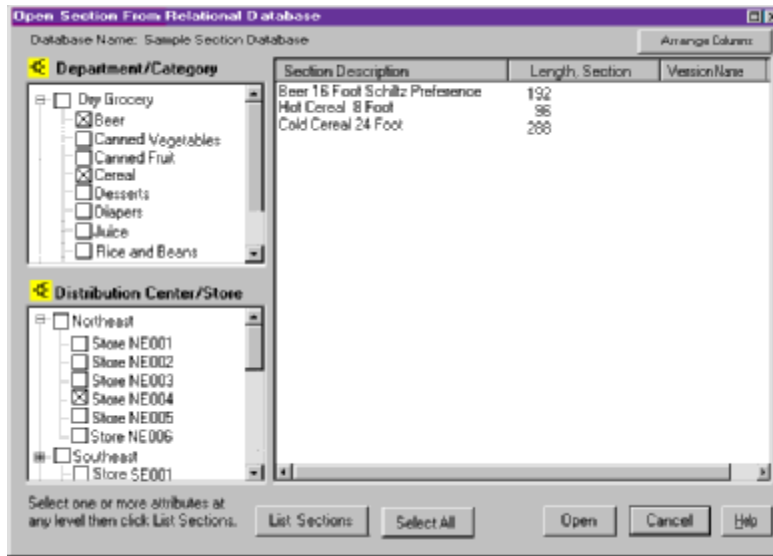
# **APOLLO 9.0**

**RELATIONAL  
SECTION  
DATABASE  
SQL QUERY  
GUIDE**

**How to Use  
MS Access™ to Extract  
and Modify Data Across  
Multiple Apollo Sections**

## Introduction

Apollo 9.0 provides a new capability for organizing and storing your planogram sections. You may now create relational databases for your sections and build custom hierarchies within them, such as the two hierarchies shown in the following Open dialog:



These hierarchies can represent any business structure that is meaningful to the organization of your sections, such as product categorization, store clustering or geography, or even using the attributes of the sections themselves, such as section length.

Once you have built and populated such a database using the tools provided in Apollo 9.0, you have the opportunity to modify, extract, and apply the information in ways that were never before possible to Apollo users.

This can be achieved by Standard Query Language (SQL) commands. The purpose of this guide is to help you understand the database table structure, provide practical examples, and help you construct your own SQL commands to achieve your business objectives.

### **Assumptions**

Apollo 9.0 is designed to work with Relational Section Databases built with Microsoft Access, Microsoft SQL, or Oracle using Open Database Connectivity (ODBC) technology.



















All of the internal utilities in Apollo 9.0 for creating and building Relational Section Databases use Microsoft Access, the most common and easy to use of the open database applications. Therefore, this guide assumes that you will be building and using your SQL queries using MS Access. It also assumes that you have some familiarity with MS Access and the general terms and concepts involved with its tables, queries, forms, and reports.

Apollo is shipped with a Sample Relational Section Database. You should familiarize yourself with this database as it is used in Apollo for opening sections and saving sections, and look at the various Relational Database Tools found in the File Menu to see how the hierarchies are built and associated with sections before delving into their underlying tables and relationships as described in this guide.

All examples and illustrations in this guide are based on the sample database provide with Apollo (Sample Relational Section Database.mdb file in the Relational Section Databases directory for Apollo90). It is recommended that you copy it to another folder and work with the copy as you use this guide.

### Database Table Overview

The Tables object display in MS Access for the Apollo Sample Relational Section Database presents the following list of tables:

	APPOSITION
	APPRODUCT
	APRETAILER
	APSECTION
	APSECTION_NEXTID
	APSHELF
	APSTORE
<hr/>	
	FST_AVAILABLECOLUMNS
	FST_DIMDETAIL
	FST_DIMMETADATA
	FST_DIMTYPE
	FST_FACT
	FST_HIERARCHY
	FST_LEVELDETAIL
	FST_SETTINGS
	FST_STRING
	FST_VIEWDIMXREF
	FST_VIEWMETADATA

Except for APSECTION\_NEXTID, the tables starting with “AP” are the traditional Apollo section data tables. These contain all the information about products, their positions on shelves, the shelves, and the section used to generate planograms and other section-specific views in Apollo. All of these tables except APRETAILER, APSTORE, and APSECTION\_NEXTID are tied together by the Section ID, which is unique to every section in the database, and provides a common link to the hierarchies.

APRETAILER and APSTORE are traditional Apollo data types, but as retailers and stores actually represent collections of sections, these tables are now used in support of Retailer and Store hierarchies. Their role will be described below in the context of hierarchies rather than sections.

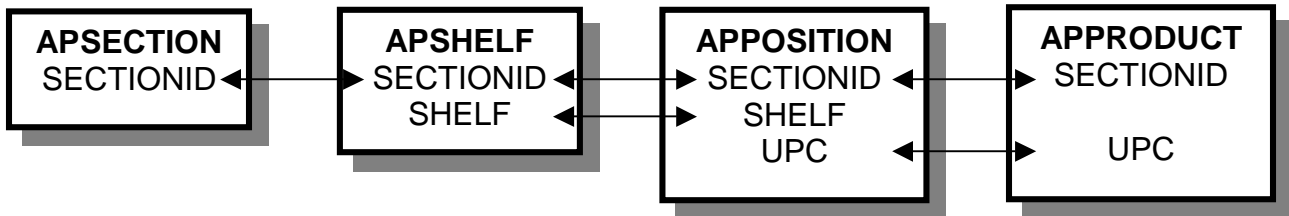
APSECTION\_NEXTID is a newly added reference table for controlling the assignment of Section IDs, and does not contain actual section data.

The remaining eleven tables, all starting with “FST\_”, are new. They manage and support the hierarchies that you build for organizing your sections. These hierarchies are displayed whenever you open, save, append, or delete a section in a Relational Section Database. “FST” is shorthand for “forest,” the internal development term for processes that work with multiple hierarchies, also known as “trees.”

Fortunately, not all eleven tables are necessary to understand or use when performing database queries. The most relevant AP and FST tables are described in detail below.

### Section Tables (AP)

The four AP tables that contain section data have their own natural relational structure. In order to compose the most powerful and dynamic queries, it is important to understand how these tables relate to each other as well as to the hierarchies that contain the sections. The following diagram shows the tables, their relationships, and the field names that are used to link them together:

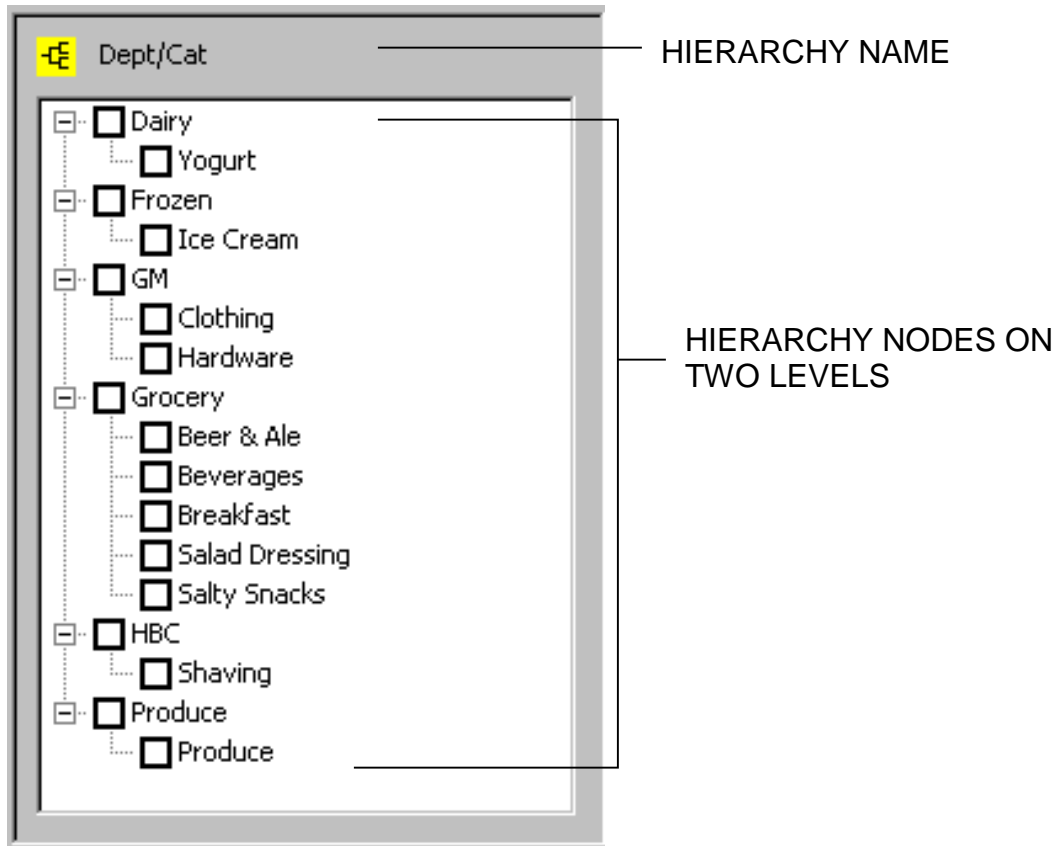


All the data in these tables is available for your database queries. Any field may be accessed individually or in the context of its relationships. For a list of all the fields in any of these tables use the table's Design View in MS Access. The Apollo data fields are also described in the Apollo Data Fields topics in the Apollo HelpStation Reference chapter.

## Hierarchy Tables (FST)

The FST tables handle section hierarchy data. Several of these are overhead or reference tables that do not deal directly with hierarchy identification, content, or relationships. The most relevant FST tables are FST\_DIMMETADATA, FST\_STRING, FST\_HIERARCHY, and FST\_FACT. Also, the Apollo Store and Retailer tables APSTORE and APRETAILER come into play here as they are used to construct User Defined hierarchies.

The best way to approach and understand what these tables are and how they relate to each other and to sections is to start with a hierarchy as it is seen in Apollo. The following is the hierarchy view window for the Department/Category hierarchy in the Sample Relational Section Database:



This hierarchy view shows the hierarchy name (Dept/Cat) and values for departments and categories on two levels. We will call these values nodes. The icon also tells us that this is a User Defined hierarchy.



## APOLLO 9.0 RELATIONAL SECTION DATABASE QUERY GUIDE

### FST\_DIMMETADATA

To have a query use a particular hierarchy, you may know its name but you will need to know its number as well. Within the database the reference number for identifying hierarchies is a field called DIMID, which stands for Dimension ID (DIM is the internal development term used for hierarchies). The DIMID for any hierarchy can be found in the FST\_DIMMETADATA table (the dimension metadata table, i.e. table of information about the hierarchies):

	DimId	DimTypeId	DimName	ReqOnSave	ReadOnly	MultiSelOnBro	MultiS
	1	1	Dept/Cat	1	0	1	
	2	1	Retailer/Store	0	0	1	

According to the table, the DIMID for the hierarchy named Dept/Cat is 1. This table contains many other settings for hierarchies, but for the purposes of basic query writing, finding the DIMID is its most important role.

### FST\_STRING

In addition to the hierarchy's name are the nodes that make up the hierarchy itself. In the sample Dept/Cat hierarchy, these are the department and category names. For those names that have been manually entered when setting up a User Defined hierarchy, the names are kept in the FST\_STRING table (the table where string or text values are kept).

STRID	VALUE
2	Dairy
3	Frozen
4	GM
5	Grocery
6	HBC
7	Produce
8	Yogurt
9	Produce
10	Shaving
11	Beer & Ale
12	Breakfast
13	Salad Dressing
14	Salty Snacks
15	Hardware
16	Clothing
17	Ice Cream
18	Beverages

DEPARTMENT NAMES

CATEGORY NAMES

This table does not include:

- Names derived from the **Store** or **Retailer** tables (as in the sample hierarchy for Retailer/Store),
- Names used in **Section Attribute** hierarchies, which are based on Apollo section or shelf field names.

## APOLLO 9.0 RELATIONAL SECTION DATABASE QUERY GUIDE

---

The STRID field is a simple sequential numbering, and does not denote a nodes place in a hierarchy.

Unlike FST\_DIMMETADATA, the FST\_STRING table will be commonly used in your query commands to call for or show hierarchy values as they pertain to the section data you are finding or changing.

### APRETAILER and APSTORE

A User Defined hierarchy may include levels that are populated from tables rather than manually entered. Apollo 9.0 provides two such tables: APRETAILER and APSTORE. Queries using or getting the names of stores and retailers need to refer to these tables for the text NAME of the nodes.

#### APRETAILER

T	NAME	NUM	RETAILER	DESCRIPTION
	DEFAULT			1
	American Markets		101	
	ACME Convenience		102	
	SuperDuper Stores		103	

In the case of retailers, the connection is made by a direct use of the retailer number in the field RETAILER.

#### APSTORE

T	NAME	STORE	RETAILER	ID	D
0	DEFAULT	1	1	1	
0	Northeast	1010	101	2	
	Mid-West	1020	101	3	
	South	1030	101	4	
	Gas Station	2010	102	5	
	Urban	2020	102	6	
	Suburban	2030	102	7	
	Chicago	3010	103	8	
	St. Louis	3020	103	9	
	Dallas	3030	103	10	

In the case of stores, the connection is made by an indirect use of the store number in the field STORE. Your queries must link store numbers to the number in the ID field to get to the name. This will be illustrated in the examples. Note that in many cases the store number is sufficient as that is visible in the hierarchy as well as the name.

These tables are populated using the Hierarchy Editor tool or the Load Single Sections tool.

## APOLLO 9.0 RELATIONAL SECTION DATABASE QUERY GUIDE

---

### FST\_HIERARCHY

The FST\_HIERARCHY table contains the actual structure of each hierarchy. Each hierarchy node is assigned a unique reference number called the HIERID. This table includes the hierarchy identifier (DIMID) that each HIERID belongs to. Each HIERID also points to its parent HIERID in the hierarchy using the PARENT field (the root or top level nodes have a PARENT value of zero). Finally, there is the pointer to the text name for the hierarchy node (FKEY), which may refer to FST\_STRING, APRETAILER or APSTORE depending on the source of the node name.

	HIERID	DIMID	PARENT	FKEY
	1	1	0	2
	2	1	0	3
	3	1	0	4
	4	1	0	5
	5	1	0	6
	6	1	0	7
	7	1	1	8
	8	1	6	9
	9	1	5	10
	10	1	4	11
	11	1	4	12
	12	1	4	13
	13	1	4	14
	14	1	3	15
	15	1	3	16
	16	1	2	17
	17	1	4	18
	18	2	0	101
	19	2	0	102
	20	2	0	103
	21	2	18	2
	22	2	18	3
	23	2	18	4
	24	2	19	5
	25	2	19	6
	26	2	19	7
	27	2	20	8
	28	2	20	9
	29	2	20	10

For DIMID 1 (Dept/Cat) the FKEY values are all found in the FST\_STRING table.

For DIMID 2 (Retailer/Store), the FKEY values come from the APRETAILER table (RETAILER=101-103) and the APSTORE table (ID=2-10).

## APOLLO 9.0 RELATIONAL SECTION DATABASE QUERY GUIDE

---

### FST\_FACT

The FST\_FACT table connects hierarchies to sections. Here the Section ID as seen in Apollo is called OBJECTID. Sections are assigned to a hierarchy node such as the category Cereal, represented by the HIERID number. And each node is associated with a hierarchy such as Dept/Cat, represented by the DIMID number. There are 12 sections in the Sample Relational Section Database, but these have 102 associations with hierarchy nodes. The following illustration shows all the section assignments in the Dept/Cat hierarchy (DIMID 1) and some from the Retailer/ Store hierarchy (DIMID 2).

	OBJECTID	HIERID	DIMID	OBJECTTYPE
	13	7	1	APSECTION
	2	8	1	APSECTION
	6	8	1	APSECTION
	4	9	1	APSECTION
	7	10	1	APSECTION
	10	11	1	APSECTION
	3	12	1	APSECTION
	11	13	1	APSECTION
	8	14	1	APSECTION
	1	15	1	APSECTION
	9	16	1	APSECTION
	15	17	1	APSECTION
	2	21	2	APSECTION
	3	21	2	APSECTION
	4	21	2	APSECTION
	6	21	2	APSECTION
	8	21	2	APSECTION
	9	21	2	APSECTION
	10	21	2	APSECTION
	11	21	2	APSECTION
	13	21	2	APSECTION
	15	21	2	APSECTION
	7	22	2	APSECTION

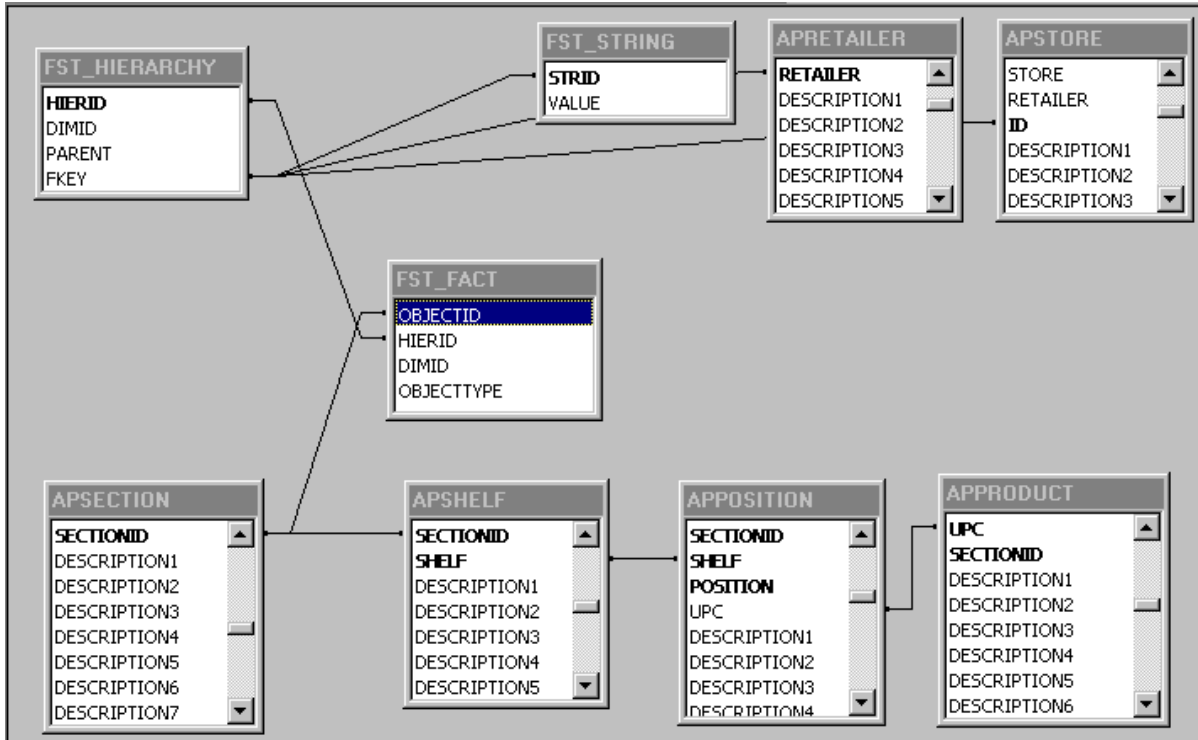
### HIERARCHY TABLE SUMMARY

In summary, here is a helpful reference chart for understanding the role of the six important FST hierarchy tables and fields that you will use in your SQL queries:

<b>FST_DIMMETADATA</b>	Get the reference number for any hierarchy name in the Relational Section Database to use in queries. DIMID = Identification number of a hierarchy DIMNAME = Hierarchy name shown in Apollo
<b>FST_STRING</b>	Provides text names for manually entered User Defined hierarchy nodes. Include this table in queries that need to either return hierarchy node names or perform an action on sections belonging to a given hierarchy node. STRID = Identification number of a node name VALUE = Name of a manually entered User Defined hierarchy node.
<b>APRETAILER</b>	Provides text names for User Defined hierarchy nodes derived from the Retailer table. Include this table in queries that need to either return retailer names or perform an action on sections belonging to a given retailer. RETAILER = Retailer number NAME = Name of a RETAILER hierarchy node
<b>APSTORE</b>	Provides text names for User Defined hierarchy nodes derived from the Store table. Include this table in queries that need to either return store names or perform an action on sections belonging to a given store. STORE = Store number as used in Apollo ID = Store ID used in FST_HIERARCHY NAME = Name of a STORE hierarchy node
<b>FST_HIERARCHY</b>	Connects hierarchies with their nodes and points to text names for the nodes. HIERID = Identification number of a hierarchy node DIMID = Identification number of a hierarchy PARENT = The HIERID that the node belongs to FKEY = Reference to the node text name (STRID, RETAILER, or store ID)
<b>FST_FACT</b>	Connects hierarchies and their nodes to Apollo sections. HIERID = Identification number of a hierarchy node DIMID = Identification number of a hierarchy OBJECTID = Section ID

## Entity Relationship Diagram

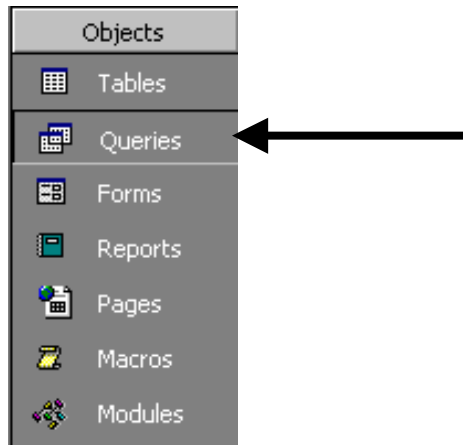
The following Access diagram shows how the FST tables described above relate to each other for the purpose of making logical connections for SQL queries to get or modify data in AP tables for sections assigned to the hierarchies in a Relational Section Database:



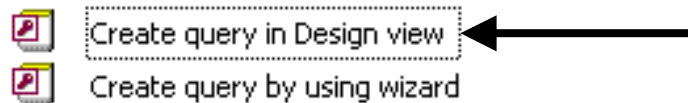
## Building SQL Queries in MS Access

The queries shown here are written in straight SQL language. This is easier for working with interchangeable sub-queries that perform different types of tasks. This also makes it easy for you to create your own queries by cutting and pasting sub-query segments and entering your own variables. Also, queries written this way may be transportable to MS SQL and Oracle relational section databases.

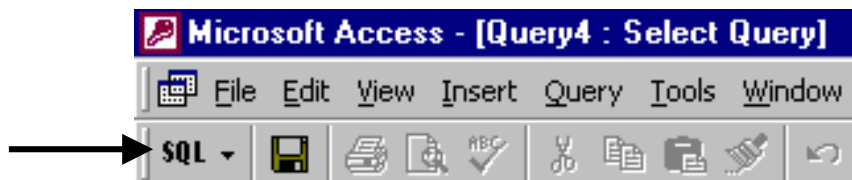
1. From MS Access open Sample Relational Section Database.mdb.
2. Under Objects, select Queries.



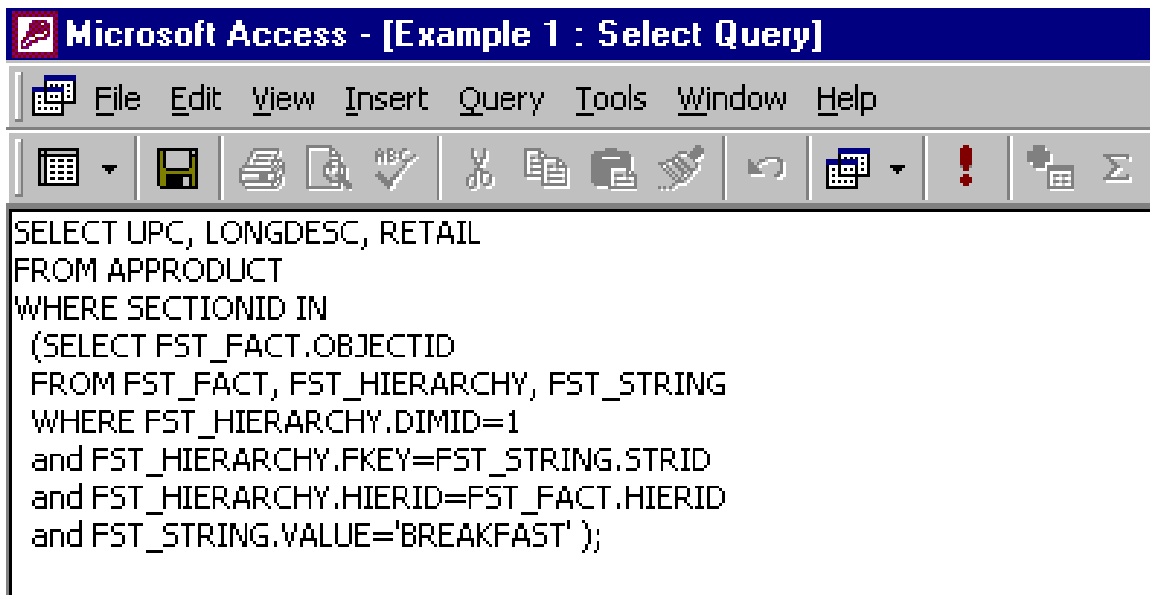
3. Select "Create Query in Design View."



4. Close the Show Table dialog.
5. Click on SQL in the toolbar or select SQL View from the View Menu.



The SQL view is a large white space where SQL commands may be written, cut or copied and pasted, like a blank page in a word processor. By default, Access starts a query with "SELECT;". Queries may be named and saved, and will be listed in the Query Objects display. When a query is run it displays the results in table form, which can be exported to spreadsheet, text, or other database formats using Export from the File Menu. The examples provided on the following pages may be pasted or copied into your Access SQL query space and run against your copy of the Sample Relational Section Database.



The screenshot shows the Microsoft Access interface. The title bar reads "Microsoft Access - [Example 1 : Select Query]". The menu bar includes File, Edit, View, Insert, Query, Tools, Window, and Help. The toolbar contains icons for grid view, save, print, zoom, undo, redo, and other standard functions. The main area displays the following SQL query:

```
SELECT UPC, LONGDESC, RETAIL
FROM APPRODUCT
WHERE SECTIONID IN
(SELECT FST_FACT.OBJECTID
FROM FST_FACT, FST_HIERARCHY, FST_STRING
WHERE FST_HIERARCHY.DIMID=1
and FST_HIERARCHY.FKEY=FST_STRING.STRID
and FST_HIERARCHY.HIERID=FST_FACT.HIERID
and FST_STRING.VALUE='BREAKFAST' );
```

**NOTE on Quotation Marks in Examples:** When copying the SQL examples from the following examples into MS Access from MS Word, it may be necessary to reenter the single quotes in Access to provide the appropriate form.

**NOTE on the UPC Field:** To retrieve or update data for a specific UPC, if the number has leading zeros be sure to include them in queries against the APPRODUCT table, but do not include them in queries against the APPOSITION Table.

### Queries That Retrieve Data

Queries that return the contents of specified fields in section tables for sections that belong to given hierarchy nodes are useful for reporting and analysis, and they can be used as sub-queries for performing queries that modify data.

The structure of the following query examples supports the most common type of informational query you will use: retrieving product or section data from some set (collection or “project”) of sections as defined in your hierarchies. The query is structured to provide the following instructions:

1. The section data fields to be returned and the section tables they come from.
2. The hierarchies and nodes that contain the sections to be interrogated.

This is written as two SELECT statements where the first takes section data from the sections retrieved with the second. More SELECT statements are used if more than one hierarchy is involved.

### Example 1 – Listing Product Details for Sections in a Category

Narrative Form:

List the UPC, Long Description, and Price for the products in all the sections in the Breakfast category of the Dept/Cat hierarchy.

SQL Form:

```
SELECT UPC, LONGDESC, RETAIL FROM APPRODUCT
WHERE SECTIONID IN
(SELECT FST_FACT.OBJECTID
FROM FST_FACT, FST_HIERARCHY, FST_STRING
WHERE FST_HIERARCHY.DIMID=1
and FST_HIERARCHY.FKEY=FST_STRING.STRID
and FST_HIERARCHY.HIERID=FST_FACT.HIERID
and FST_STRING.VALUE='BREAKFAST' );
```

Using the Sample Relational Section Database, the results should return 90 records and look like:

UPC	LONGDESC	RETAIL
1313000054	NABISCO SPO	3.41
1600063830	GENERAL MILL	2.79
1600065810	GENERAL MILL	2.22
1600065850	GENERAL MILL	3.04
1600065890	GENERAL MILL	4.15
1600066210	GENERAL MILL	3.18
1600068790	GENERAL MILL	2.99
3000004340	QUAKER OATS	3.59
3000006320	QUAKER OAT I	3.07
3000006340	QUAKER OAT I	2.91
3000006500	QUAKER CAP	3.03
3000006560	QUAKER CAP	3.03
3000006610	QUAKER CAP	3.03
3800000501	KELLOGGS RIK	1.76
3800000510	KELLOGGS RIK	2.2
3800000810	KELLOGGS RA	2.58

Commentary:

“SELECT UPC, LONGDESC, RETAIL FROM APPRODUCT” tells Access to display a list of UPC codes, Long Descriptions, and Prices from the Product table.

“WHERE SECTIONID IN (SELECT FST\_FACT.OBJECTID” relates the SECTIONID field in the product table to the OBJECTID in the Fact table.

“FROM FST\_FACT, FST\_HIERARCHY, FST\_STRING” establishes the hierarchy tables to be used.

“WHERE FST\_HIERARCHY.DIMID=1” limits the query to the Dept/Cat hierarchy.

“and FST\_HIERARCHY.FKEY=FST\_STRING.STRID” connects the hierarchy node ID’s with their text names.

“and FST\_HIERARCHY.HIERID= FST\_FACT.HIERID” joins the Hierarchy and Fact tables on the node ID.

“and FST\_HIERARCHY.FKEY=FST\_STRING.STRID and FST\_STRING.VALUE = 'BREAKFAST' );” joins the Hierarchy and String tables, and limits the query to the Breakfast category.

As you can see, much of this is “boilerplate” that can be used over and over. The parts of this query that are variables are UPC, LONGDESC, RETAIL, APPRODUCT, DIMID=1, and BREAKFAST.

### Example 2 - Listing Qualified Product Data for a Category

Narrative Form:

List the UPC, Long Descriptions, and Movement for the products in the sections in the Breakfast category where the Movement is less than 100.

SQL Form:

```
SELECT UPC, LONGDESC, REALMOVEMENT FROM APPRODUCT
WHERE APPRODUCT.REALMOVEMENT < 100
AND SECTIONID IN
(SELECT FST_FACT.OBJECTID
FROM FST_FACT, FST_HIERARCHY, FST_STRING
WHERE FST_HIERARCHY.DIMID=1
and FST_HIERARCHY.HIERID=FST_FACT.HIERID
and FST_HIERARCHY.FKEY=FST_STRING.STRID
and FST_STRING.VALUE='BREAKFAST');
```

Using the Sample Relational Section Database, the results should return 4 records and look like:

UPC	LONGDESC	REALMOVEMENT
3800005231	KELLOGGS ASSORTED ASSORTED ASSORTED 6.375OZ	97.43
3800013700	KELLOGGS LOW FAT GRANOLA REGULAR WHL GRAIN OAT & WHT CLUSTE 14OZ	74.77
3800005231	KELLOGGS ASSORTED ASSORTED ASSORTED 6.375OZ	97.43
3800013700	KELLOGGS LOW FAT GRANOLA REGULAR WHL GRAIN OAT & WHT CLUSTE 14OZ	74.77

Commentary:

Including an extra qualification was simply a matter of adding “APPRODUCT.REALMOVEMENT < 100” to the main WHERE statement.

### Example 3 - Listing Product Data for Hierarchy Combinations

Narrative Form:

List the UPC and Long Descriptions for products in the Breakfast category section that are assigned to the Chicago store (#3010).

SQL Form:

```
SELECT UPC, LONGDESC
FROM APPRODUCT
WHERE SECTIONID IN
  (SELECT FST_FACT.OBJECTID FROM FST_FACT, FST_HIERARCHY,
  FST_STRING
  WHERE FST_HIERARCHY.DIMID=1
  and FST_HIERARCHY.HIERID=FST_FACT.HIERID
  and FST_HIERARCHY.FKEY=FST_STRING.STRID
  and FST_STRING.VALUE='BREAKFAST')
  AND SECTIONID IN
  (SELECT FST_FACT.OBJECTID FROM FST_FACT,
  FST_HIERARCHY, APSTORE
  WHERE FST_HIERARCHY.DIMID=2
  and FST_HIERARCHY.HIERID=FST_FACT.HIERID
  and FST_HIERARCHY.FKEY=APSTORE.ID
  and APSTORE.STORE=3010);
```

Using the Sample Relational Section Database, the results should return 45 records and look like:

UPC	LONGDESC
1313000054	NABISCO SPOON SIZE SHREDDED WHEAT REGULAR WHOLE WHEAT BIS 17.2OZ
1600063830	GENERAL MILLS HONEY FROSTED WHEATIES HONEY WHEAT FLAKE 14.75OZ
1600065810	GENERAL MILLS TOTAL REGULAR WHEAT AND BROWN RICE FLAKE 8OZ
1600065850	GENERAL MILLS TOTAL REGULAR WHOLE GRAIN WHEAT FLAKE 12OZ
1600065890	GENERAL MILLS TOTAL REGULAR WHOLE GRAIN WHEAT FLAKE 18OZ
1600065960	GENERAL MILLS TOTAL CORN FLAKES REGULAR CORN FLAKE 10OZ
1600066110	GENERAL MILLS WHEATIES REGULAR WHOLE GRAIN WHEAT FLAKE 12OZ
1600066210	GENERAL MILLS WHEATIES REGULAR WHOLE GRAIN WHEAT FLAKE 18OZ
1600068790	GENERAL MILLS MULTI GRAIN CHEERIOS REGULAR MULTI GRAIN RING 16OZ
3000004340	QUAKER OATS CAPTAIN CRUNCH CRUNCH BERRIES OATS & CORN 19OZ
3000006320	QUAKER OAT LIFE CINNAMON WHOLE GRAIN OAT SQUARE 15.5OZ
3000006340	QUAKER OAT LIFE REGULAR WHOLE GRAIN OAT SQUARE 15OZ

Commentary:

Combining two or more hierarchies in a query is simply a matter of adding SELECT statements for the different hierarchies. In this case, DIMID=2 refers to

## APOLLO 9.0 RELATIONAL SECTION DATABASE QUERY GUIDE

---

the Retailer/Store table. This can also be done for combining different nodes in the same hierarchy.

Note that a different command is needed to get to the store numbers (FST\_HIERARCHY.FKEY=APSTORE.ID) where APSTORE is the location of the store number and name, rather than the FST\_STRING table because stores, like retailers, are not manually entered nodes.

The parts of this query that are variables are UPC, LONGDESC, APPRODUCT, DIMID=1, BREAKFAST, DIMID=2, STORE=3010.

Using the Sample Relational Section Database, this query should return 45 records.

### Queries That Modify Data

Queries that change the contents of specified fields in section tables may be more commonly performed without needing to limit the query by hierarchy nodes. For example, any specific product attribute update such as price, cost, or perhaps a dimensional change due to new packaging can be applied directly to the APPRODUCT table without requiring further qualification by hierarchy (Example 4). Otherwise, SELECT statements would be used to define subsets of products that require updating when specific UPC's are not used (Example 5).

The data modification query is written with UPDATE and SET statements. SELECT statements are added when the update is to be applied to sections associated with specific hierarchy nodes.

#### CAUTION WHEN WORKING WITH CALCULATED FIELDS

1. Some numeric fields are calculated internally by Apollo.

A calculated value can be modified with an SQL command, but would be inconsistent with the values that are used to derive the calculation. And once Apollo opens the database, the field would be restated based on its formula.

2. Some numeric fields are used by Apollo to derive calculated fields.

If you use an SQL command to modify a value that is used by Apollo to derive a calculated field (e.g. PROFIT is derived from ADJUSTED MOVEMENT, SELLING PRICE, and COST), the calculated field is not updated unless the database is reopened in Apollo.

**RECOMMENDATION:** Use SQL queries to update only those fields that are not calculated by Apollo (see APPENDIX A – DIRECT FIELDS), then open Apollo to reset the calculated fields before doing any further queries.

See APPENDIX B for lists of the calculated fields by table.

### Example 4 – Updating a Product Price for Specific Products

Narrative Form:

Update the Price for three Kraft salad dressings (Cucumber UPC=2100064490; Rancher Choice UPC=2100064362; Catalina UPC=2100064484) to \$1.15.

SQL Form:

```
UPDATE APPRODUCT SET RETAIL = '1.15'  
WHERE UPC='2100064490'  
OR UPC= '2100064362'  
OR UPC= '2100064484';
```

There should be a warning message that three rows will be updated, and no other results displayed.

Commentary:

“UPDATE APPRODUCT” establishes the table to be updated.

“SET RETAIL='1.15'” sets the new value for the price field.

“WHERE UPC='2100064490';” identifies the record (row) to be updated.

Running an update query returns a warning prompt indicating the number of rows to be updated. No record listing is given.

### Example 5 – Assign a Uniform Value to Sections in a Category

Narrative Form:

Update Plot Description 1 with the category name for all sections in the Salty Snacks category.

SQL Form:

```
UPDATE APSECTION
  SET PLOTDESC1='Salty Snacks'
WHERE SECTIONID IN
  (SELECT FST_FACT.OBJECTID
   FROM FST_FACT, FST_HIERARCHY, FST_STRING
   WHERE FST_HIERARCHY.DIMID=1
   and FST_HIERARCHY.FKEY=FST_STRING.STRID
   and FST_HIERARCHY.HIERID=FST_FACT.HIERID
   and FST_STRING.VALUE='SALTY SNACKS' );
```

There should be a warning message that one row will be updated, and no other results displayed.

Commentary:

“UPDATE APSECTION” establishes the table to be updated.

“SET PLOTDESC1='Salty Snacks'” sets the new value for the Plot Description 1 field.

“WHERE SECTIONID IN” indicates that the APSECTION’s SECTIONID field is to be matched with the sections returned from the SELECT statement below it, which has the same structure as the SELECT statements used for retrieving data.

In the case of the Sample Relational Database, there is only one section in the Salty Snacks category.

### Example 6 – Refresh Product Data from an External Source

Narrative Form:

Update item Movement in the APPRODUCT table with the movement supplied in an external comma separated variable (csv) file for every UPC in the csv file.

External File:

Create a text file with the following contents and save as STOREUPCMVT.csv. The three variables represent store numbers, UPC, and movement numbers.

```
1010, 2100012041, 34
1020, 2100012068, 22
1030, 2100012072, 11
2010, 2100012643, 9
2020, 2100012644, 33
2030, 2100012647, 47
3010, 2100012651, 4
3020, 2100012665, 12
3030, 2100025272, 5
```

In Access, import the csv file using File/Get External Data > Import. In the Import Text Wizard, accept the input as delimited, with commas, in a new table. Name Field1 to STORE as Text; name Field 2 to UPC as Text; name Field 3 to MVT as Double. Let Access add a primary key field (ID). Name the table STOREUPCMVT.

SQL Form:

```
UPDATE APPRODUCT
INNER JOIN STOREUPCMVT ON APPRODUCT.UPC = STOREUPCMVT.UPC
SET APPRODUCT.REALMOVEMENT = STOREUPCMVT.MVT;
```

There should be a warning message that nine rows will be updated, and no other results displayed.

Commentary:

“INNER JOIN” links the new table with APPRODUCT on the UPC field.

Updates can also be done directly from external files using the File/Get External Data > Link Tables option.

Use SELECT statements as shown in previous examples to limit the update by store or category or any other hierarchy that you have set up.

**APPENDIX A– DIRECT FIELDS**

These Direct Fields are not calculated by Apollo and are ODBC mappable. They may be refreshed via SQL queries and will not be overwritten when opening Apollo.

**TABLE 1 – SECTION DIRECT FIELDS**

DESCRIPTION	FIELD NAME
Allow Float	ALLOWFLOAT
Overhang Peg/Free Hand Items	ALLOWOVERHANG
Overlap Peg/Free Hand Items	ALLOWOVERLAP
Arbitrary	ARBITRARY
Background	BACKGROUND
Date Created	CREATEDATE
Crunch Mode	CRUNCH
DecoHeight	DECOHEIGHT
DecoText	DECOTEXT
DecoWidth	DECOWIDTH
DecoXpos	DECOXPOS
DecoYpos	DECOYPOS
Depth, Section	DEPTH
Section Desc 01 – 50	DESCRIPTION01 – 50
First Notch	FIRSTNOTCH
Foreground	FOREGROUND
Height, Section	HEIGHT
Increment	INCREMENT
LogoPath	LOGOPATH
Section Meas 01 – 50	MEASURE01 – 50
Merch. Method, Section	MERCH
Message Line1	MESSAGELINE1
Message Line2	MESSAGELINE2
Message Line3	MESSAGELINE3
Message Line4	MESSAGELINE4
Message Line5	MESSAGELINE5
Section Name	NAME
Notch Spacing	NOTCH
Number of Shelves	NUMSHELVES
Plot Desc. 1, Section	PLOTDESC1
Plot Desc. 2, Section	PLOTDESC2
Plot Desc. 3, Section	PLOTDESC3
Reserved5	RESERVED5
Reserved6	RESERVED6
Section	SECTION
Section Desc (Short)	SECTIONDESC
Section ID	SECTIONID

## APOLLO 9.0 RELATIONAL SECTION DATABASE QUERY GUIDE

---

ShowDeco	SHOWDECO
Variable Height Shelves	STAGGEREDSHELVES
Standard Shelf Length	STANDARDLENGTH
Traffic Flow	TRAFFIC
Date Last Updated	UPDATEDATE
Time Last Updated	UPDATETIME
Upright Spacing	UPRIGHT
UprightBackground	UPRIGHTBACKGROUND
UprightForeground	UPRIGHTFOREGROUND
User1, Section	USER1
User2, Section	USER2
Version	VERSION
Version Name	VERSIONDESC
Length, Section	WIDTH
X-Position, Section	XPOS
Y-Position, Section	YPOS
Z-Position, Section	ZPOS

**TABLE 2 – SHELF DIRECT FIELDS**

DESCRIPTION	FIELD NAME
AutoCalcFronts	AUTOCALCFRONTS
Background	BACKGROUND
Bracket Depth	BRACKETDEPTH
Bracket Height	BRACKETHEIGHT
Bracket Increment	BRACKETINCREMENT
Bracket Slope	BRACKETSLOPE
Bracket Thickness	BRACKETTHICKNESS
Shelf Desc 01 – 50	DESCRIPTION01 – 50
Shelf Display Number	DISPLAYNUM
Divider Increment	DIVINC
Divider Thickness	DIVTHICK
Edge	EDGE
Fit Status	FITSTATUS
Foreground	FOREGROUND
Grill Height	GRILLHEIGHT
Grill Thickness	GRILLTHICKNESS
Max Merch. Depth	MAXMERCHANTDEPTH
Shelf Meas 01 – 50	MEASURE01 – 50
Max Merch. Height	MERCHANTHEIGHT
Notch Number	NOTCHNUMBER
Rotation	RESERVED1
Reserved3	RESERVED3
Reserved4	RESERVED4
Advanced Fixtures	RESERVED5
Reserved6	RESERVED6

## APOLLO 9.0 RELATIONAL SECTION DATABASE QUERY GUIDE

---

Shelf Number	SHELF
Depth, Shelf	SHELFDEPTH
Shelf Type	SHELFTYPE
Length, Shelf	SHELFWIDTH
Slope	SLOPE
Crunch Mode	SPREADMODE
Thickness	THICKNESS
Desc1 (Shelf)	USERDESC1
Desc2 (Shelf)	USERDESC2
Desc3 (Shelf)	USERDESC3
Desc4 (Shelf)	USERDESC4
Num1 (Shelf)	USERNUM1
Num2 (Shelf)	USERNUM2
Num3 (Shelf)	USERNUM3
X-Incr.	XINC
X-Peg Right	XPEGEND
X-Peg Left	XPEGSTART
X-Position, Shelf	XPOS
Y-Incr.	YINC
Y-Peg Top	YPEGEND
Y-Peg Bottom	YPEGSTART
Y-Position, Shelf	YPOS
Z-Position, Shelf	ZPOS

**TABLE 3 – POSITION DIRECT FIELDS**

DESCRIPTION	FIELD NAME
Position Desc 01 – 10	DESCRIPTION01 – 10
Edge	EDGE
Fit Status	FITSTATUS
Position Meas 01 – 30	MEASURE01 – 30
Merch. Method	MERCHANDISINGTYPE
Orientation	ORIENTATION
Peg Length	PEGLENGTH
Peg Type	PEGTYPE
Hanger Width	RESERVED1
Hanger Height	RESERVED2
Reserved5	RESERVED5
SalesGrowth	SALESGROWTH
Suggested Facings	SUGGESTEDFACINGS
Hanger Adj.	TAGNUMBER
Upc	UPC
Facings	XFACINGS
X-Gap	XGAP
X-Position	XPOS

Fronts High	YFACINGS
Y-Gap	YGAP
Y-Position	YPOS
Z-Gap	ZGAP

**TABLE 4 – PRODUCT DIRECT FIELDS**

DESCRIPTION	FIELD NAME
Case Cost Adjustment	ADJTOCASECOST
Warehouse Arrive As	ARRIVEAS
Avg. CasesDSDShipment	AVRNOCASESDSDSH
Brand	BRAND
Case Cost	CASECOST
Case Depth	CASEDEPTH
Case Height	CASEHEIGHT
Casepack	CASEPACK
Case Weight	CASEWEIGHT
Case Width	CASEWIDTH
Category	CATEGORY
Character	CHAR
Color	COLOR
Component	COMPONENT
Description 01 – 30	DESCRIPTION01 – 30
Dist. Method	DISTRIBUTION
DPCA	DPCA
Movement, Expected	EXPECTEDMOVEMENT
Finger Space	FINGER
Long Description	LONGDESC
Manufacturer	MANUFACTURER
Max Cases	MAXCASES
Max DOS	MAXDAYS
Max Facings	MAXFACING
Max Fronts	MAXFRONTS
Max Fronts Deep	MAXFRONTSDEEP
MaxHeight	MAXHEIGHT
Max Layovers	MAXLAYOVERS
Max Units	MAXUNITS
Measure 001 – 110	MEASURE001 – 110
Min Cases	MINCASES
Min DOS	MINDAYS
Min Facings	MINFACING
Min Units	MINUNITS
Mvt. Adj. Factor	MOVEMENTADJ
Multipack	MULTIPACK
Nesting	NESTING
Net Days Credit	NETDAYS CREDIT

## APOLLO 9.0 RELATIONAL SECTION DATABASE QUERY GUIDE

---

Cases per Pallet	NOCASESPALLET
Inner Packs per Case	NOINNERPACKS
Trays per Case	NOTRAYSCASE
Overhang	OVERHANG
Package Type	PACKAGE
Cash Discount	PAYDISCOUNT
Plot Desc. 1	PLOTDESC1
Plot Desc. 2	PLOTDESC2
Depth	PRODDEPTH
Height	PRODHEIGHT
Width	PRODWIDTH
Rack Opening Height	RACKOPENHT
Movement	REALMOVEMENT
Replenish Method	REPLENISHMETHOD
Restock	RESTOCK
Retail	RETAIL
Short Description	SCREENDESC
SectionId	SECTIONID
SectionShare	SECTIONSHARE
Service	SERVICE
Share	SHARE
Size	SIZE
Stockcode	STOCKCODE
Store Price Method	STOREPRICEMETHOD
Store Receive Unload	STORERECUNLOAD
Subcategory	SUBCATEGORY
Symbol	SYMBOL
UOM	UOM
UPC	UPC
Description A – J	USERDESCA – J
Measure A – J	USERMEASUREA – J
Variance	VARIANCE
VAT	VAT
Warehouse Delivery	WARERECEIVE
X-Peghole Offset	XPEGHOLE
Y-Peghole Offset	YPEGHOLE

**TABLE 5 – RETAILER DIRECT FIELDS**

DESCRIPTION	FIELD NAME
Address 1	ADDRESS1
Address 2	ADDRESS2
City	CITY
Contact	CONTACT
Retail Desc 01 – 50	DESCRIPTION01 – 50
Retail Meas 01 – 50	MEASURE01 – 50

Retailer Name	NAME
Number of Stores	NUMSTORES
Phone	PHONE
Retailer	RETAILER
State	STATE
Zip Code, Retailer	ZIP

**TABLE 6 – STORE DIRECT FIELDS**

DESCRIPTION	FIELD NAME
ACV	ACV
Address 1, Store	ADDRESS1
Address 2, Store	ADDRESS2
City, Store	CITY
Cluster	CLUSTER
Contact	CONTACT
Depth, Store	DEPTH
Store Desc 01 – 50	DESCRIPTION01 – 50
District	DISTRICT
Height, Store	HEIGHT
Store Meas 01 – 50	MEASURE01 - 50
Store Name	NAME
Phone, Store	PHONE
State, Store	STATE
Store	STORE
User1, Store	USER1
User2, Store	USER2
User3, Store	USER3
User4, Store	USER4
Length, Store	WIDTH
Zip Code, Store	ZIP

**APPENDIX B – CALCULATED FIELDS**

Apollo calculates these fields. If they are modified via SQL queries they will be overwritten when opening Apollo. NOTE: There are no calculated fields in the Retailer and Store tables.

**TABLE 1 – CALCULATED SECTION FIELDS**

DESCRIPTION	FIELD NAME
Cub In/Cm Avail, Section	AVAILABLECUBIC
Lin In/Cm Avail, Section	AVAILABLELINEAR
Sq In/Cm Avail, Section	AVAILABLESQUARE
Avg. Cases, Section	AVERAGECASES
Avg. Facings, Section	AVERAGEFACINGS
Capacity, Section	CAPACITY
Cases, Sect.	CASES
Cubic Unit Multiplier	CUBICMULTIPLIER
Cubic Throughput, Section	CUBICTHROUGHPUT
Number Days In Week	DAYSINWEEK
DPP, Section	DPP
Facings, Section	FACINGS
Highest Cases, Section	HICASES
Highest DOS, Section	HIDAYS
Highest Facings, Section	HIFACINGS
Inv-Retail, Section	INVATRETAIL
Inv-Cost, Section	INVENTORY
Linear Unit Multiplier	LINEARMULTIPLIER
Lowest Cases, Section	LOWCASES
Lowest DOS, Section	LOWDAYS
Lowest Facings, Section	LOWFACINGS
Movement, Section	MOVEMENT
SKUs, Section	NUMSKUS
Profit, Section	PROFIT
ROI, Section	ROI
Sales, Section	SALES
Square Unit Multiplier	SQUAREMULTIPLIER
Cub In/Cm Used, Section	USEDCUBIC
Lin In/Cm Used, Section	USEDLINEAR
Sq In/Cm Used, Section	USED SQUARE

**TABLE 2 – CALCULATED SHELF FIELDS**

DESCRIPTION	FIELD NAME
Cub In/Cm Avail., Shelf	AVAILABLECUBIC
Lin In/Cm Avail., Shelf	AVAILABLELINEAR
Sq In/Cm Avail., Shelf	AVAILABLESQUARE
Avg. Cases, Shelf	AVERAGECASES
Avg. Facings, Shelf	AVERAGEFACINGS
Capacity, Shelf	CAPACITY
Cases, Shelf	CASES
Cubic Throughput, Shelf	CUBICTHROUGHPUT
DPP, Shelf	DPP
Facings, Shelf	FACINGS
Highest Cases, Shelf	HICASES
Highest DOS, Shelf	HIDAYS
Highest Facings, Shelf	HIFACINGS
Inv-Retail, Shelf	INVATRETAIL
Inv-Cost, Shelf	INVENTORY
Lowest Cases, Shelf	LOWCASES
Lowest DOS, Shelf	LOWDAYS
Lowest Facings, Shelf	LOWFACINGS
Merchandisable Depth	MERCHDEPTH
Movement, Shelf	MOVEMENT
SKUs, Shelf	NUMSKUS
Profit, Shelf	PROFIT
ROII, Shelf	ROII
Sales, Shelf	SALES
Height, Shelf	SHELFHEIGHT
Cub In/Cm Used, Shelf	USEDUBIC
Lin In/Cm Used, Shelf	USEDLINEAR
Sq In/Cm Used, Shelf	USED SQUARE

**TABLE 3 – CALCULATED POSITION FIELDS**

DESCRIPTION	FIELD NAME
Margin, Actual	ACTUALMARGIN
Movement, Adjusted	ADJUSTEDMOVEMENT
Back Room Stock	BACKROOMSTOCK
Capacity	CAPACITY
Capacity/Cubic	CAPACITYPERCUBIC
Capacity/Linear	CAPACITYPERLINEAR
Capacity/Square	CAPACITYPERSQUARE
Cases	CASES
Margin, Contrib	CONTRIBTOMARGIN
Cubic In/Cm	CUBIC
Days of Supply	DAYSOFSUPPLY
Movement, Distributed	DISTMOVEMENT
DPP	DPP
DPP %	DPPPERCENT
DPP/Cubic	DPPPERCUBIC
DPP/Linear	DPPPERLINEAR
DPP/Square	DPPPERSQUARE
DPP/Unit	DPPPERUNIT
GMROI	GMROI
Inv-Cst/Cubic	INVCOSTPERCUBIC
Inv-Cst/Linear	INVCOSTPERLINEAR
Inv-Cst/Square	INVCOSTPERSQUARE
Inv-Cst	INVENTORY
Inv-Rtl	INVENTORYATRETAIL
Inv-Rtl/Cubic	INVRETPERCUBIC
Inv-Rtl/Linear	INVRETPERLINEAR
Inv-Rtl/Square	INVRETPERSQUARE
Layovers	LAYOVERS
Layovers Deep	LAYOVERSDEEP
Lin In/Cm	LINEAR
Lost Sales	LOSTSALES
Movement/Cubic	MOVEMENTPERCUBIC
Movement/Linear	MOVEMENTPERLINEAR
Movement/Square	MOVEMENTPERSQUARE
MultiCapacity	MULTICAPACITY
MultiHand Orient	MULTIORIENT
Normal Stock	NORMALSTOCK
%Capacity/Section	PCTCAPACITYPERSECT

## APOLLO 9.0 RELATIONAL SECTION DATABASE QUERY GUIDE

---

%Capacity/Shelf	PCTCAPACITYPERSHF
%Cubic/Section	PCTCUBICPERSECT
%Cubic/Shelf	PCTCUBICPERSHF
%DPP/Section	PCTDPPPERSECT
%DPP/Shelf	PCTDPPPERSHF
%Inv-Cst/Section	PCTINVCOSTPERSECT
%Inv-Cst/Shelf	PCTINVCOSTPERSHF
%Linear/Section	PCTLINEARPERSECT
%Linear/Shelf	PCTLINEARPERSHF
%Movement/Section	PCTMOVEMENTPERSECT
%Movement/Shelf	PCTMOVEMENTPERSHF
%Profit/Section	PCTPROFITPERSECT
%Profit/Shelf	PCTPROFITPERSHF
%Sales/Section	PCTSALESPERSECT
%Sales/Shelf	PCTSALESPERSHF
%Square/Section	PCTSQUAREPERSECT
%Square/Shelf	PCTSQUAREPERSHF
Position	POSITION
Profit	PROFIT
Profit/Cubic	PROFITPERCUBIC
Profit/Linear	PROFITPERLINEAR
Profit/Square	PROFITPERSQUARE
Profit/Unit	PROFITPERUNIT
ROII	ROII
Safety Stock	SAFETYSTOCK
Sales	SALES
Sales/Cubic	SALESPERCUBIC
Sales/Facing	SALESPERFACING
Sales/Linear	SALESPERLINEAR
Sales/Square	SALESPERSQUARE
Square In/Cm	SQUARE
Turns	URNS
Weeks of Supply	WEEKSOFSUPPLY
X-End Position	X_ENDPOS
X-Dimension	XDIM
X-Peghole Position	XPEGHOLEPOS
X-Peg Number	XPEGNUM
Y-End Position	Y_ENDPOS
Y-Dimension	YDIM
Y-Peghole Position	YPEGHOLEPOS
Y-Peg Number	YPEGNUM
Z-End Position	Z_ENDPOS
Z-Dimension	ZDIM

Fronts Deep	ZFACINGS
Z-Position	ZPOS

**TABLE 4 –CALCULATED PRODUCT FIELDS**

DESCRIPTION	FIELD NAME
Cost	COST
Margin	MARGIN
Selling Price	SELLINGPRICE