

WELCOME

Accessing the TRAMS Database Using SQL
"This class will go in depth on the structure of the TRAMS database and how to use SQL statements to extract data for reporting purposes."

Dan Palley
CTO, TRAMS

*For further information and individualized assistance,
please visit one of our Study Hall sessions.*

Accessing TRAMS Using SQL

Presentation Breakdown

1. Introduction to SQL
2. Overview of the TRAMS Database
3. Sample Queries

1. Introduction To SQL

- What Is A Database?
- What is SQL?
- Components of an SQL Database.
- SQL Syntax for Querying Data
- SQL and Interbase
- SQL Alternatives

1. Introduction To SQL

- What Is A Database?
 - A Database is an organized collection of information in computerized format.
 - A Database is generally implemented in a Database Management System (DBMS), i.e., Interbase.

1. Introduction To SQL

■ What Is SQL?

- SQL Stands for Structured Query Language
- SQL is a standard interactive and programming language for getting information from and updating a database.
- SQL is the de facto method for working with databases.
- Almost all DBMS's support SQL in some form.

1. Introduction To SQL

■ Components of an SQL Database

SQL Databases are made up of Fields, Records, Tables, Keys, and Indexes.

- ❖ A Field is a single data element.
- ❖ A Record is a grouping of related data elements (fields).
- ❖ A Table is a collection of Records of the same type.
- ❖ A Key is a combination of one or more Fields that uniquely identify each Record.
- ❖ An Index is specified per table and can be made up of one or more fields.
- ❖ A Database is made up of one or more Tables.

1. Introduction To SQL

■ SQL Syntax for Querying Data

SQL Requests for Data are called Queries,
which return Rows or Results

- ❖ A Query result may contain zero or more rows.
- ❖ A row is simply a list of field values.
- ❖ A row may correspond to multiple records in multiple tables if a JOIN is used.

1. Introduction To SQL

■ SQL Syntax for Querying Data

Table and Field Aliases

- ❖ Table Aliases are used to clarify the SQL.
- ❖ Table Aliases are also used to prevent ambiguity when selecting from different tables that have the same field name or when selecting from multiple instances of the same table.
- ❖ Field Aliases are used mainly to name the columns in the Results.
- ❖ Examples:
Table1 AS T1, Field1 F1

1. Introduction To SQL

- SQL Syntax for Querying Data
The Basic SELECT/FROM Statement

```
Select T1.Field1, T1.Field2 From Table1 T1
```

T1 is a Table Alias for Table1.

T1.Field1 and T1.Field2 are Fields of Table T1.

1. Introduction To SQL

- SQL Syntax for Querying Data
Filtering Results Using the WHERE Clause

- Equality Operators

- ❖ Field1 = value1
- ❖ Field1 >= value1
- ❖ Field1 < value1

- String Operators

- ❖ STARTING WITH

- P.NameUpper Starting with 'SMITH'
- Uses an index if present.
- Is case-sensitive.

- ❖ CONTAINING

- Upper(A.Address1) CONTAINING 'GILLMAN ST'
- Will not ever use an index.
- Is case-sensitive but can be gotten around using UPPER().

1. Introduction To SQL

■ SQL Syntax for Querying Data

Filtering Results Using the WHERE Clause, Cont'd.

- Specifying Multiple Values In A Single Expression
 - ❖ BETWEEN value1 AND value2
 - IssueDate Between '1/1/05' and '1/31/05'
 - ❖ IN (value1, value2, value3)
 - ProfileType_LinkCode In ('I','C')
 - ❖ EXISTS/NOT EXISTS (Sub-Select)
 - Exists (Select InvoiceNo From Invoice Where Client_Linkno=5)
 - ❖ Using Boolean operators (AND/OR) to specify multiple expressions
 - Field1=5 AND Field2='TEST'AND Field3='01/01/05'
 - (Field1=5 Or Field2='TEST') And Field3='01/01/05'

If No WHERE clause is specified, then the query will return results for all records in the table(s) specified in the FROM clause.

1. Introduction To SQL

■ SQL Syntax for Querying Data

Sorting Results Using the ORDER BY Clause

- If you need to see the results in a specific order, then you must specify an ORDER BY.
- You can specify multiple fields to use as the ORDER and each field can specify Ascending or Descending.
- Fields in the ORDER BY clause do not need to appear in the SELECT clause.
- Specifying an ORDER BY can slow down queries that return large numbers of results; only use when necessary.

1. Introduction To SQL

- SQL Syntax for Querying Data

Querying multiple Tables using the JOIN/ON syntax.

```
Select T1.Field1, T2.Field2  
From Table1 T1  
Join Table2 T2 On T1.Field3=T2.Field3_LinkNo
```

1. Introduction To SQL

- SQL Syntax for Querying Data

Eliminating Duplicate Results Using the DISTINCT Keyword.

```
Select Distinct T1.Field1, T1.Field2, T1.Field3  
From Table1 T1
```

- Be careful about using Distinct on queries that return large numbers of rows, since it will slow the query down.

1. Introduction To SQL

- SQL Syntax for Querying Data

Aggregate Functions (Sum, Min, Max) and the GROUP BY Clause

- Select Sum(B.TotalFare) From Booking B
- Select B.SubmitTo_LinkCode, Sum(B.TotalFare) From Booking B Group By B.SubmitTo_LinkCode

1. Introduction To SQL

- SQL Syntax for Querying Data

Combining Multiple Select Statements Together using the UNION Keyword

- Each SELECT statement must specify the same type and number of fields.

```
Select field1, field2 from table1  
Union  
Select field3, field4 from table2
```
- UNION by itself will remove any duplicate rows from the resulting set.
- Like Distinct, UNION can slow down queries with a large number of results.
- UNION ALL will retain duplicate rows (and run faster as a result).

1. Introduction To SQL

- SQL and Interbase
 - SQL is the native protocol for querying and updating an Interbase database.
 - Most 3rd party applications (i.e. Crystal Reports) will use an ODBC Driver (EasySoft) to send SQL statements to Interbase.
 - Interbase SQL tools
 - ❖ IB Console
 - ❖ Crystal Reports/MS Office
 - ❖ TRAMS IB Query Utility
- SQL Alternatives
 - Crystal Database Wizard
 - MS Office Query Wizard

2. Overview of the TRAMS Database

There Are Six Major Table Groupings. Some Tables are common to both TBO and CB+ and others are only for TBO or CB+.

- Profiles
- Invoices
- Payments
- ResCards (CB+ Only)
- Activities (CB+ Only)
- G/L (TBO Only)

2. Overview of the TRAMS Database

■ Profiles

- All profile types (Leisure, Corporate, Vendor, Agent and Other) are stored in the Profile table.
- Addresses are stored in the Address table.
- Communications (Phone, Fax, Email) are in the Communications table.
- Most other tables point back to the Profile table
- Invoices point to a Client Profile
- Bookings point to a Vendor Profile.
- AgentBkgs point to an Agent Profile.
- Payments can point to any type of Profile.

2. Overview of the TRAMS Database

■ Invoices

- An Invoice can contain one or more Bookings.
- Each Booking can contain one or more Segments.
- UDID and AgentBkg entries are below the Booking level.
- You can check if a booking is voided by checking ClientPayStatus_LinkCode='V'.

2. Overview of the TRAMS Database

■ Payments

- All payment types (Received, Made, Deposit, Withdrawal) are stored in the Payment table.
- Payments that apply to invoices create one or more PayDtl records so that a single payment can link to multiple invoices.

■ G/L (TBO Only)

- G/L Account information is stored in the GL Table; G/L Account Balances are in the GLDetail table (for multiple balance sheets).
- Journal Entries are stored in the JE and JEDtl (line item) tables.

2. Overview of the TRAMS Database

■ ResCards (CB+ Only)

- A ResCard can contain one or more ResCardReservations.
- A ResCardReservation can contain one or more ResCardProviders and one or more ResCardSegments.

■ Activities (CB+ Only)

- All Activity Types (Notes, Reminders, Mailers) Are stored in the Activity table.

3. Sample Queries

- Profile Queries
- Invoice/Booking Queries
- Payment Queries
- CityPair Queries
- Getting a single record match in a master/detail (one-to-many) situation

3. Sample Queries

- Profile Queries
 - All Client Profiles

```
Select P.ProfileNo, P.Name
From Profile P
Where P.ProfileType_LinkCode In ('I','C')
```
 - By Profile Group

```
Select P.Profileno, P.Name
From Marketing M
Join Profile P On P.ProfileNo=M.Profile_LinkNo
Where M.MarketingValue='CORPORATE'
```

3. Sample Queries

■ Profile Queries

□ By Marketing Code (CB+ Only)

- ❖ Profiles where each of 3 Marketing Codes is selected

```
select p.profileno, p.name
from profilemarketing pm1
join profilemarketing pm2 on
pm2.profile_linkno=pm1.profile_linkno
join profilemarketing pm3 on
pm3.profile_linkno=pm1.profile_linkno
join profile p on pm1.profile_linkno=p.profileno
where pm1.code_linkno=4 and
pm2.code_linkno=10 and
pm3.code_linkno=31
```

3. Sample Queries

■ Profile Queries

□ By Marketing Code (CB+ Only)

- ❖ Profiles where any of 3 Marketing Codes is selected

```
select p.profileno, p.name
from profilemarketing pm1
join profile p on p.profileno=pm1.profile_linkno
where pm1.code_linkno=4
union
select p.profileno, p.name
from profilemarketing pm2
join profile p on p.profileno=pm2.profile_linkno
where pm2.code_linkno=10
union
select p.profileno, p.name
from profilemarketing pm3
join profile p on p.profileno=pm3.profile_linkno
where pm3.code_linkno=31
```

- ❖ Note: The UNION keyword by itself will eliminate duplicate rows.

3. Sample Queries

■ Profile Queries

- Profiles With Invoice Activity In The Last Year

```
Select P.ProfileNo, P.Name  
From Profile P  
Left Join Invoice I on  
I.Client_LinkNo=P.ProfileNo And  
I.IssueDate>='Today'-365  
where P.ProfileType_LinkCode In ('I','C') And  
I.InvoiceNo Is Null
```

3. Sample Queries

■ Profile Queries

- Profiles With No Invoice Activity In The Last Year

```
Select P.ProfileNo, P.Name  
From Profile P  
Left Join Invoice I on  
I.Client_LinkNo=P.ProfileNo And  
I.IssueDate>='Today'-365  
where P.ProfileType_LinkCode In ('I','C') And  
I.InvoiceNo Is Null
```

3. Sample Queries

■ Invoice/Booking Queries

□ Client Sales By Travel Type

```
Select P.ProfileNo, P.Name ClientName, T.TravelType, Sum(B.TotalFare)
From Invoice I
  Left Join Booking B On B.Invoice_LinkNo=I.InvoiceNo
  Left Join Profile P On P.ProfileNo=I.Client_LinkNo
  Left Join TravelType T On T.TravelTypeNo=B.TravelType_LinkNo
Where I.IssueDate Between '1/1' And 'today' And
      B.ClientPayStatus_LinkCode <> 'V'
Group By P.ProfileNo, P.Name, T.TravelType
```

We have to group by ProfileNo since we may have duplicate profile names.

3. Sample Queries

■ Invoice/Booking Queries

□ Preferred Vendor Sales By Travel Category

```
Select P.ProfileNo, P.Name VendorName, TC.TravelCategory, Sum(B.TotalFare)
From Invoice I
  Left Join Booking B On B.Invoice_LinkNo=I.InvoiceNo
  Left Join Profile P On P.ProfileNo=B.Vendor_LinkNo
  Left Join TravelType TT on TT.TravelTypeNo=B.TravelType_LinkNo
  Left Join TravelCategory TC On
TC.TravelCategoryNo=TT.TravelCategory_LinkNo
Where I.IssueDate Between '1/1' And 'today' And
      B.ClientPayStatus_LinkCode <> 'V' And
      P.PreferredVendor='Y'
Group By P.ProfileNo, P.Name, TC.TravelCategory
```

3. Sample Queries

■ Payment Query

C/C Payment Information By Credit Card Type. This example demonstrates how to look up a credit card type based on the first 2 digits of the credit card number.

```
Select Y.PaymentDate, Y.Amount, P.Name, T.Code, Y.CkCCNo
From Payment Y
Join Profile P On P.ProfileNo=Y.Profile_LinkNo
Join CCNo_Lookup L On L.CCNo=Cast(f_mid(f_fixccno(Y.CkCCNo)||'00',0,2) As
Integer)
Join CCTypes T On T.TypeNo=L.CCType_LinkNo
Where Y.PaymentDate Between '1/1' And 'today' And
Y.PayMethod_LinkNo In (3,5) And
T.TypeNo=0
```

3. Sample Queries

■ CityPair Query

Each Air Booking can be made up of one or more City Pairs (Legs) and each City Pair can be made up of one or more Segments. A City Pair includes Origin and Final Destination information as well as Fare and FareBasis for all of the corresponding segments.

```
Select CP.DepartCity , CD.CityName ,CP.ArriveCity , CA.CityName, CP.Fare,
CP.FareBasis,
B.ValidAL, B.PassengerName, B.StartingTicketNo, I.InvoiceNumber
From CityPair CP
Left Join Invoice I On I.InvoiceNo=CP.Invoice_LinkNo
Left Join Booking B On B.BookingNo=CP.Booking_LinkNo
Left Join City CD On CP.DepartCity =CD.City
Left Join City CA On CP.ArriveCity =CA.City
Left Join Profile VP On VP.ProfileNo=B.Vendor_LinkNo
Where CP.DepartDateTime between '1/1' And 'today' And
I.Paystatus_LinkCode <> 'V' And
I.InvoiceType_LinkCode='S' And
B.ClientPayStatus_LinkCode <> 'V'
```

3. Sample Queries

- Getting a single record match in a master/detail (one-to-many) situation.
 - Profile Example Using Joins
 - ❖ The Profile entity is made up of numerous tables since a given profile can have multiple addresses, phone numbers, etc.
 - ❖ Many times, you need profile data like name, address, city/state/zip, and phone but only want one result row. To get a single row result, you need to specify just the primary address, phone, etc.

3. Sample Queries

- Getting a single record match in a master/detail (one-to-many) situation.
 - Profile Example Using Joins, Cont'd.

```
Select P.ProfileNo, P.ProfileType_LinkCode, P.Name,  
A.Address1, A.Address2, A.City, A.State, A.Zip,  
CP.CommValue, CF.CommValue, CE.CommValue  
From Profile P  
Left Join AddrInstance AI On AI.Profile_LinkNo=P.ProfileNo And  
AI.AddrType_LinkNo=1  
Left Join Address A On A.AddressNo=AI.Address_LinkNo  
Left Join Communication CP On Cp.Profile_LinkNo=ProfileNo And  
CP.CommType_LinkNo=2 And CP.IsPrimary = 'Y'  
Left Join Communication CF On CF.Profile_LinkNo=ProfileNo And  
CF.CommType_LinkNo=1 and CF.IsPrimary = 'Y'  
Left Join Communication CE On CE.Profile_LinkNo=ProfileNo And  
CE.CommType_LinkNo=3 and CE.IsPrimary = 'Y'
```

We use Left Outer Joins so that profiles that don't contain address or phone information will still return results (the missing fields will be blank).

3. Sample Queries

- Getting a single record match in a master/detail (one-to-many) situation.
 - Invoice Example Using A Combination Of Joins and Sub-Selects
 - ❖ Each invoice can have multiple bookings so how do I run a query that shows one result per invoice where I don't know which booking to include or where I want to include certain information from a single booking and summarized information from all bookings?

3. Sample Queries

- Getting a single record match in a master/detail (one-to-many) situation.
 - Invoice Example Using A Combination Of Joins and Sub-Selects, Cont'd.
 - ❖ Example: Show the invoice number, issue date, invoice type, first passenger name (passenger name from the first booking on the invoice), and sum of Booking total fares.

```
Select I.InvoiceNumber, I.IssueDate, I.InvoiceType_LinkCode, B.PassengerName,  
(Select Sum(TotalFare) From Booking Where Invoice_Linkno=I.Invoiceno) TotalFare  
From Invoice I  
Left Join Booking B on B.BookingNo=(Select Min(BookingNo+0) From Booking Where  
Invoice_Linkno=I.Invoiceno)
```

The "+0" above is required to work around an optimizer bug. You can omit it, but the query will not be as fast.

References

TRAMS Crystal Reports/Database Newsgroup

<news://news.trams.com/general.crystalreporting>

TRAMS Data Dictionaries

<http://www.trams.com/dd/index.html>

SQL Books/Tutorials

- Teach Yourself SQL in 21 Days, 2nd Edition
<http://members.tripod.com/er4ebus/sql/ch01.htm>
- SQL For Smarties, Joe Celko

Thank You

Check your personal agenda for
the location of your next class,
or stop by the Tech U
registration desk for directions.

*For further information and individualized assistance,
please visit one of our Study Hall sessions.*