



Welcome to Travel Experts Travel Agency. Let us book your next vacation!

TRAVEL EXPERTS TRAVEL AGENCY

Database Design, Creation, and Administration

Kenny Vigar

kvigar@gmail.com

Travel Experts Travel Agency

Table of Contents

Business Proposal.....	Section 1
Executive Summary.....	1
Requirements/Scope.....	2
Budget Considerations.....	4
Project Schedule	9
Deliverables.....	10
Performance and Tuning.....	14
System and Database Analysis	18
Next Steps	20
Appendix (Schedule, ERD, TIC)	23
 Requirements Documentation	 Section 2
Project Definition	1
Project Scope	2
Business Requirements.....	3
Data Requirements	4
Technical Requirements.....	5
Budget Plan.....	6
Appendix (Interview Questions, Organized Notes)	7
 Table Instance Charts	 Section 3
 Database Design Recipe.....	 Section 4
Database Blueprint	1
Overview of Environment	2
Create Database Recipe.....	2
Verification.....	4
Create Database Script.....	5
Appendix (Tablespace Verification Script)	9
Appendix (Parameter File)	12
Appendix (Entity Relationship Diagram (full))	13

Travel Experts Travel Agency

Create Tables and Data Load	Section 5
Cleanup Code	1
Adhoc Table Load.....	5
Data Clean Up with SQL	10
Create Tables	25
Employee	25
EmployeeDetail	26
Salary	27
Customer	32
CustomerDetail.....	33
CreditCard.....	34
CustomerComm.....	39
Fee	43
Destination	44
TravelClass.....	47
Affiliation	49
Supplier.....	50
SupplierDetail	52
Product	56
Itinerary	60
Payment.....	64
Sale	66
Commission	68
Verification.....	69
SQLLDR Control and Log Files	90
Create Users	Section 6
Cleanup Code	1
Profile Creation.....	2
Verify Profile Creation.....	6
Role Creation.....	9
Verify Role Creation	15
Create Users	18
Verify Users	25
Auditing Considerations	30
Network Strategy	Section 7
Prepare User Account for Testing	1
Verify Testing User Account.....	8
Server Side Network Configuration	12
Verify Network Configuration	13
Test Results.....	16
TNSNAMES.ORA, LISTENER.ORA, SQLNET.ORA	19, 21, 22

Travel Experts Travel Agency

Data Warehouse and Analysis	Section 8
Background	1
External Data Load (External Table, SQLLDR).....	2
Fact Table Partitioning	3
Create Fact Table	4
Merge Data to Fact Table.....	5
Fact Table Tuning (Material Views, Outline)	6
Create Dimension Tables	7
Data Analysis	
Data Analysis (Excel Pivots, Graphs)	13
Data Analysis (SUMSUM)	16
Marketing Recommendations	17
Backup and Recovery	Section 9
Background	1
Gather and Prepare Files.....	2
Archive Log Mode	3
Setup Fast Recovery Area	4
Setup Flashback	5
Setup Recovery Manager.....	6
Setup Recovery Catalog	7
Instance Recovery	8
Backup Schedule	9
Schedule Legend	10
Disk Strategy	15
Emergency Contact Information	16
Appendix (Data File Recovery and Redo Log Recovery Examples)	17
Appendix (Queries to Find Critical Files for Backup)	27
Appendix (Linux Backup Shell Script Example)	28
Website Integration	Section 10
Benefits	1
Existing Examples.....	2
Travel Experts Web Integration.....	6
Public Pages	7
Internal Pages	9
Data Validation	11
Internal Users and Security	12

Travel Experts Travel Agency

Performance Tuning	Section 11
Designing Performance	1
Instance Tuning	2
Memory Management	3
SQL Tuning	5
Appendix (Execution Plans).....	6
Invoice and Commission Reports	Section 12
Invoice Report Sample	1
Commission Report Sample	2
Invoice Report Script.....	4
Commission Report Script.....	7
December Stakeholder Meeting	Appendix 1
April Stakeholder Meeting	Appendix 2
Project Schedule	Appendix 3
Workshop Descriptions	Appendix 4





ELCARO 2.0

DATABASE PROJECT PROPOSAL



Submitted to: Travel Experts Travel Agency

Submit Date: Friday April 28th 2017

Submitted by: ElCaro2.0 Database Development

1301 16 Ave NW

Calgary, AB T2M 0L4

ElCaro2.0 Database Development
1301 16 Ave NW
Calgary, AB T2M 0L4
403.870.9208



Travel Experts Travel Agency
1155 8th Ave S.W.
Calgary, Alberta
T2P1N3
403.271.9873

April 22 2017

Dear Karen Graham

I am writing to you to on behalf of ElCaro2.0 Database Development regarding your database project request for proposal. ElCaro2.0 has extensive experience in database creation and maintenance and we would be very excited to complete this project.

ElCaro2.0 specializes in Oracle RDBMS software implementation/installation, Extract Transform Load processes, performance tuning, disaster recovery strategies and procedure/function development. ElCaro2.0 also has extensive experience with requirement gathering, business/data warehouse analysis, and data reporting.

In addition to our data administration training, we have a combined experience of over 60 years as a team and have years of experience working in corporate environments providing efficient and knowledgeable software, hardware and analysis support to various teams including management and executive levels.

Being able to work easily with clients and colleagues, as well as quick thinking on the job are some of ElCaro2.0's greatest strengths which allow us to adapt to technology such as database systems which are always evolving.

I would be happy to meet any time to discuss this opportunity. Please feel free to contact me by email or phone at your convenience.

Thank you for your time and consideration.
Sincerely,

Kenny Vigar
403.870.9208
kvigar@gmail.com



Topics of Discussion

Executive Summary	1
Requirements Documentation.....	2
- Business Requirements	
- Data Requirements	
- Technical Requirements	
- Project Scope	3
- Budget Considerations	4
Project Schedule (condensed).....	9
Project Deliverables	10
- Entity Relationship Diagram (simplified).....	10
- Sample Invoice.....	11
- Sample Commission Report	12
- Performance and Tuning recommendations.....	14
System and Database Analysis	18
- Hardware/OS/Database Recommendations.....	18
- Database Blueprint.....	19
Next Steps.....	20
- Web Integration with Oracle Application Express.....	20
- Data Warehousing and Sample Sales analysis Reports...	22
Appendix 1	
Project Schedule (full).....	23
Appendix 2	
Entity Relationship Diagram (full).....	24
Table Instance Charts (full).....	25
Appendix 3	
TEA Invoice Script	32
TEA Commission Report Script	35



Executive Summary

Travel Experts Travel Agency (TEA) is located in Calgary Alberta and currently employs 16 travel agents and management. During preliminary discussions with management regarding this proposal's RFP, it was learned that TEA management expects 10% customer growth in the next 5 years. To accommodate these projections, TEA would like to move towards a central and secure location to store customer and sales information.

TEA has found its agents in an isolated or "siloed" situation when it comes to keeping track of customer contact information, travel/booking information and employee records. With different agents all having a preferred method of recording their data confusion can arise. If a client visits the TEA office on when their preferred agent is unavailable, there may be issues finding client's travel information. More frightening – it could be discovered the data was lost all together with no hope of recovery.

Daily business complications that will be resolved by using a database system as opposed to spreadsheets and hardcopy filing include the ability to quickly access data that was just recently updated by other employees, analysis on outstanding and paid commissions from product suppliers as well as the ability to recover any lost data or to secure it from being removed from premises.

Based off these requirements a database design built from the ground up with performance and efficiency as a top concern was developed. The design discussion includes categorizing and normalizing data to the 3rd form. Capturing only related information to its relevant entity and ensuring referential integrity with data verification tables will keep information concise and clean.

TEA's business requirements have been determined and a project scope has been provided in greater detail on page 5 of this document.

ElCaro2.0's objective is to provide a database solution that will have leading edge hardware powering the Oracle architecture the TEA Database will be running on, which will incur zero data loss and give 100% data consistency. The ability to provide accurate reporting on commissions, and sales with speed and reliability is factored into our solution from the very beginning of the database design plan straight to our planned rollout of May 2017.

ElCaro2.0 employs design and ERD specialists, analytic and reporting gurus, and data recovery/performance tuning experts. This ensures your agents have the best tools available to help TEA meet the growing need to provide world class business, leisure and affordable travel to their clients.



Requirements Documentation

No.	Requirement	Mandatory / Would Like
1	Provide an easy to use graphical interface to enter and view client data with Oracle ApEx.	M
2	Ability to waive the agency fee as per agent discretion.	M
3	Develop detailed methods to follow marketing trends and top sales agents with analytics reporting through data warehousing and custom PL/SQL and SQL scripts.	M
4	Ability to verify commission tracking. Cheques are sent, generally, 60 days after the product is used, based on the return or end date of the product booking.	M
5	Scripts to generate the invoice based on the client and booking information.	M
6	Tracking the difference between business, group or leisure travel. As well, the option to track which business users may be interested in leisure travel opportunities.	M
7	Eventually would like a web interface for the system for clients to book online.	W

Table 2.1- Business Requirements

Information Type	Requirements
Spreadsheet and Supplier DVD/Book.	<p>Consolidate all hardcopy and softcopy information to a single source (database). This will be checked for data anomalies and ‘cleaned’ before entering into the database.</p> <p>There are currently just over 10,000 individual values of data in the spreadsheet to load to the database (pre-cleanup)</p> <p>Supplier product information is also available every six months but can contain out of date information. The information on products available are in either a book (\$75) or DVD (\$300).</p>
Access and storage	<p>All users will have access to the system, but only the commission specialist will have elevated access to update supplier and cost information.</p> <p>Other users will be able to query data and enter new customer/sales records in the system.</p>
Growth	Owner expects to have a 5 – 10% client growth and the database should be able to handle that.
Information stored	<p>Client name, address and phone numbers.</p> <p>Reward or frequent flyer numbers.</p> <p>Credit card information.</p> <p>Booking number must be unique – Currently it is unique by supplier.</p> <p>Commissions may change values.</p> <p>Each Payment is a new invoice.</p>

Table 2.2 - Data Requirements



No.	Requirement
1	System will need full business hour functionality, with maintenance and upgrades occurring after hours.
2	Additional backup, reliability will be provided by an additional server on site.
3	Database will be backed up nightly as the information stored is changed daily.
4	System will be built on Oracle technologies.
5	All agents will be receiving hardware upgrades in the form of new computers, monitors, and a central server to run the database from.
6	The ability to flag records in the system as needing an update. The flag will notify agents that changes to an account are required when it is pulled up in the system.

2.3 - Technical Requirements

Scope of Project

We will be providing the following for the TEA database/GUI design project

- Requirement gathering
- Analysis of current hardware and ensure it fits the requirements or replace
- Hardware procurement
- Budgeting details
- Database design
- 'Easy to use' graphical interface design
- Data cleanup and data importing to the new database
- Database development
- 'Easy to use' graphical interface development
- Report building (market trends, top sales, commission reports)
- Develop query/customer entry scripts
- Security implementation (user accounts, database security, table privilege access)
- Database optimization and tuning
- Backup and recovery processes
- User training and user technical support (as per contract terms)
- System rollout

Out of Scope

This project does not include the following, and will need to be negotiated into the contract terms if required

- User support past the agreed date in the contract
- Web integration of system
- On call or emergency support
- Additional ElCaro2.0 employees beyond the assigned technicians (4 technicians included)
- Additional hardware or training not covered in the scope of the project (advanced SQL training, advanced report or script building)



Budget Considerations

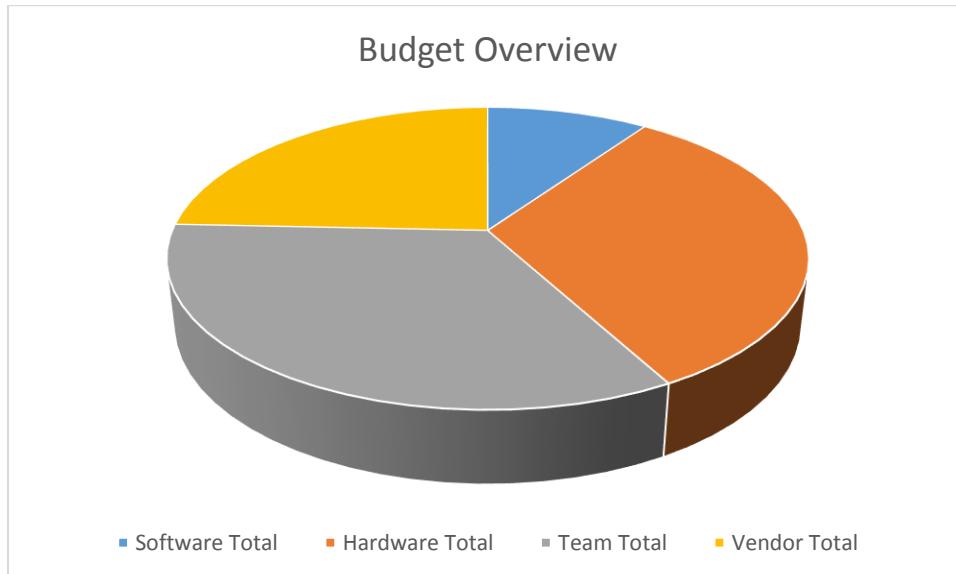


Figure 2.1 - Overall Budget Summary

TEA has provided an initial budget constraint of \$500,000 for all components of the project. This includes hardware procurements or upgrades, software, ElCaro2.0 team specialists and additional third party vendor services such as Iron Mountain backup storage, etc.

The budget has been broken down on the following pages detailing each of the categories in the above "Budget Overview" chart. Anything included in the scope documentation has been included in the category breakdowns but anything additional to the project scope was not included. This includes 3rd party website hosting and any additional after hours support.

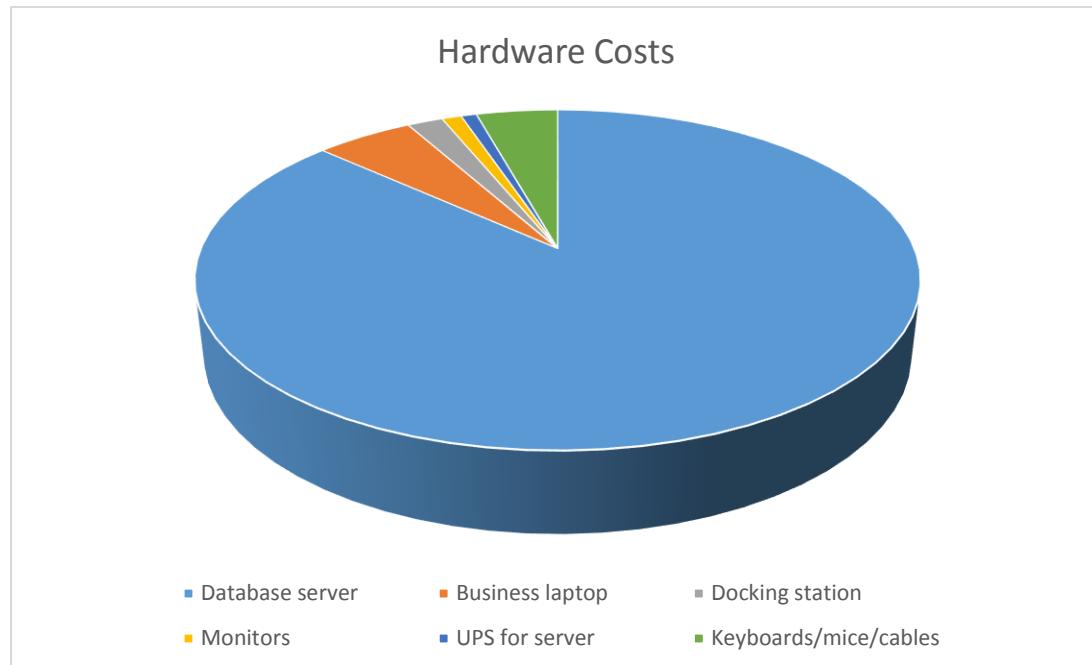


Figure 2.2 - Hardware Costs

Resource Type	Quantity	Product	Cost	Details
Hardware	2	Database server	10,000.00	ME PC V_ST3100I Server System
Hardware	10	Business laptop	599.99	Lenovo G50 80KR0015US
Hardware	10	Docking station	219.00	Targus 4K Universal Docking Station
Hardware	20	Monitors	119.99	22M38D-B 21.5in Full HD
Hardware	1	UPS for server	94.99	Back-UPS ES 550VA
Hardware	Misc.	Keyboards/mice/cables	500.00	Misc hardware for connecting and using computers

Table 2.3 - Hardware Costs

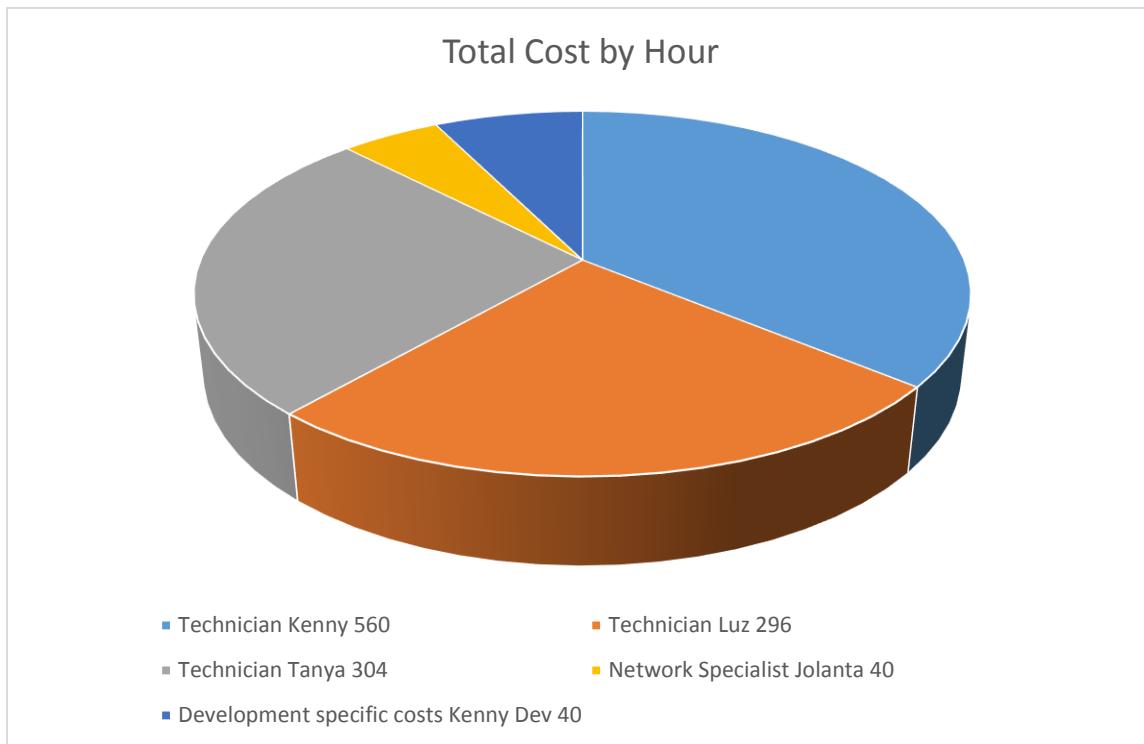
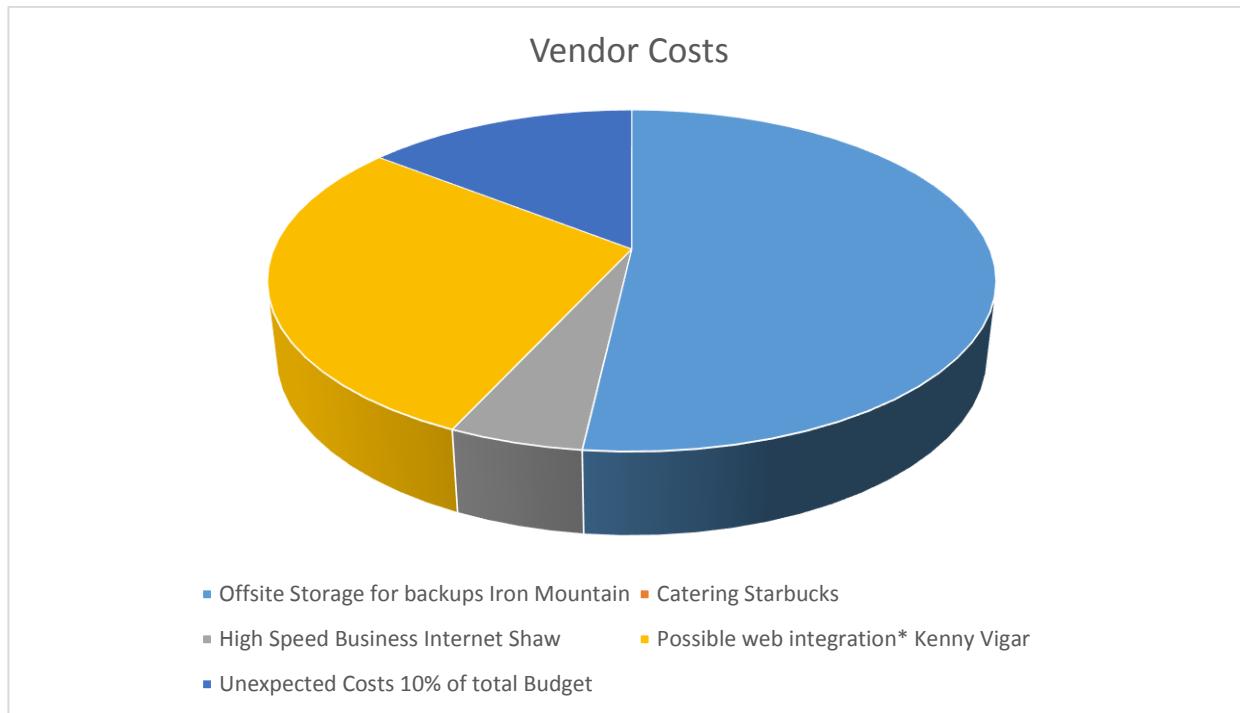


Figure 2.4 - ELCARO 2.0 Specialist Costs

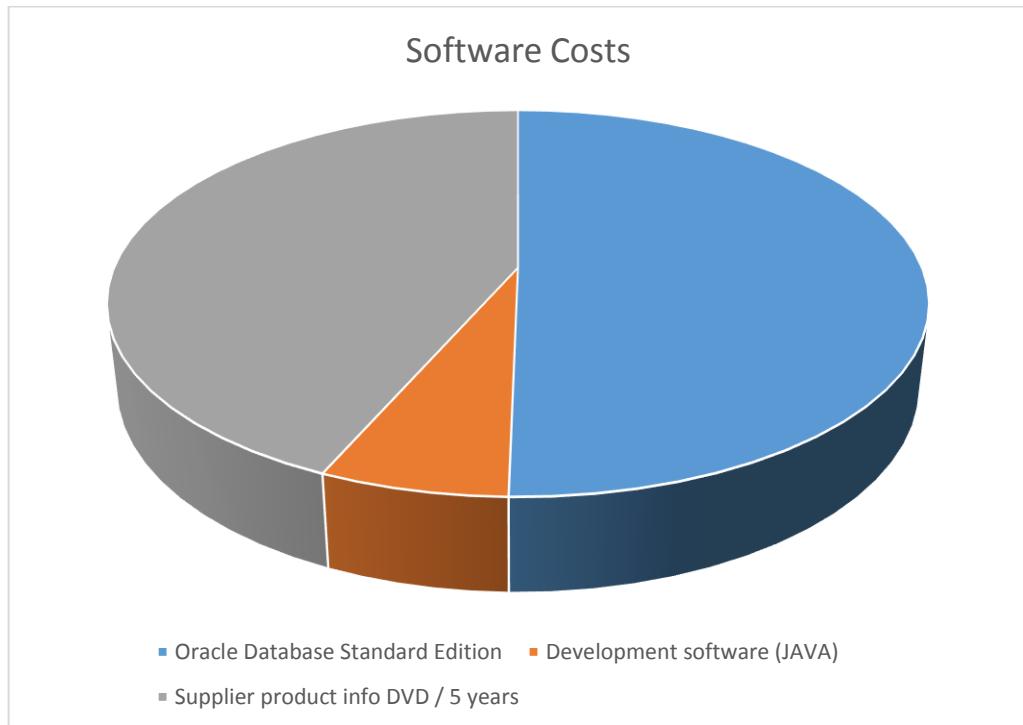
Resource Type	Name	Hours (total)	Cost	Total Cost
Technician	Kenny	560	70/hour	29,200.00
Technician	Luz	296	70/hour	20,720.00
Technician	Tanya	304	70/hour	21,280.00
Network Specialist	Jolanta	40	100/hour	4000.00
Development specific costs	Kenny – Developer Hours	40	150/hour	6000.00

2.5 - ELCARO 2.0 Specialist Costs

2.6 - *Vendor Costs*

Additional Resources	Supplier	Cost	Total Cost
Offsite Storage for backups	Iron Mountain	3000/month over 5 years	180,000.00
Catering	Starbucks		7,800 .00
High Speed Business Internet	Shaw	3500/month over 5 years	17,500.00
Possible web integration*	Kenny Vigar		100,000
Unexpected Costs	10% of total Budget		50000

2.7 - *Vendor Costs*



2.8 - Software Costs

Resource Type	Quantity	Product	Cost	Details
Software	1	Oracle Database Standard Edition	20,850.00	Oracle Software and licenses
Software	1	Development software (JAVA)	2,600.00	Software to code front end GUI
Software	60	Supplier product info DVD / 5 years	18,000.00	Supplier information to base commission estimate.

2.9 - Software Costs



Schedule Overview

A general project schedule is shown below. This is a condensed schedule – a fully detailed schedule follows in the appendix of this proposal.

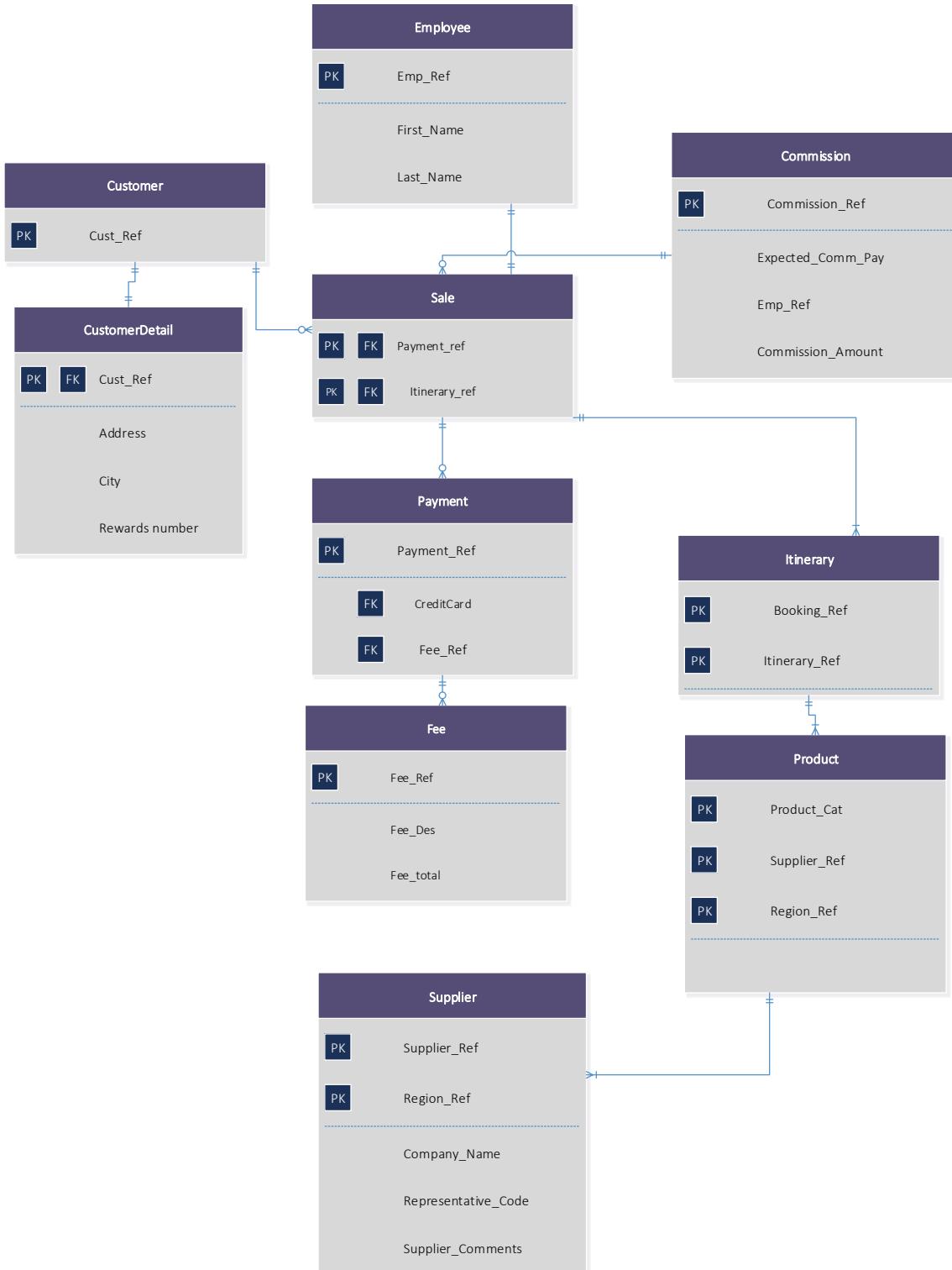
Task	Duration	Start	Finish	% Complete
Analysis	7 days	Tue 10/25/16	Wed 11/2/16	100%
Procure Hardware	1 day	Thu 11/3/16	Thu 11/3/16	100%
Design Database	11 days	Fri 11/4/16	Fri 11/17/16	100%
Meeting with Stakeholders	0 days	Tue 11/17/16	Fri 11/17/16	100%
Develop Database	20 days	Mon 11/18/16	Fri 12/16/16	100%
Meeting with Stakeholders	0 days	Tue 12/17/16	Tue 12/17/16	100%
Create Database Load Data	20 days	Tue 12/18/16	Mon 01/06/17	100%
Meeting with Stakeholders	0 days	Mon 01/06/17	Mon 01/06/17	100%
Create Users and Privileges	2 days	Wed 01/18/17	Thu 01/19/17	100%
User Privilege and Role Testing	1 day	Fri 01/27/17	Fri 01/28/17	100%
Setup Network Access to Database	3 days	Fri 01/29/17	Tue 01/31/17	100%
Extensive Network Access Testing	1 days	Sat 02/04/17	Tue 02/06/17	100%
Meeting with Stakeholders	0 days	Tue 02/07/17	Tue 02/07/17	100%
Design GUI Interface (Oracle Application Express)	10 days	Wed 02/08/17	Tue 02/21/17	100%
Design User Interface for Entering Data (GUI)	10 days	Wed 02/22/17	Tue 03/07/17	100%
Meeting with Stakeholders	0 days	Tue 03/07/17	Tue 03/07/17	100%
Database Testing	3 days	Wed 03/08/17	Fri 03/11/17	100%
Design Database Security and Audit Processes	6 days	Mon 03/13/17	Wed 03/15/17	100%
Create User Profiles, Roles, and Assign Permissions	2 days	Mon 03/13/17	Tue 03/14/17	100%
Develop User Creation Scripts	1 day	Wed 03/15/17	Wed 03/15/17	100%
Develop Front End GUI	7 days	Thu 03/16/17	Thu 03/23/17	100%
Meeting with Stakeholders	1 day	Fri 03/24/17	Fri 03/24/17	100%
Design GUI interface to load new data	6 days	Thu 03/25/17	Thu 03/31/17	100%
Test process of importing data with GUI	1 day	Fri 04/02/17	Fri 04/02/17	100%
Meeting with Stakeholders	0 days	Mon 04/05/17	Mon 04/05/17	100%
Hardware Setup	6 days	Mon 04/06/17	Mon 04/12/17	100%
Design backup and recovery strategy	6 days	Tue 04/13/17	Tue 04/19/17	100%
Meeting with Stakeholders	0 days	Wed 04/19/17	Wed 04/19/17	100%
Preform initial backup of data	2 days	Thu 04/20/17	Fri 04/22/17	100%
Preform test data restoration	2 days	Mon 04/23/17	Tue 04/24/17	100%
Meeting with Stakeholders	0 days	Tue 04/25/17	Tue 04/25/17	100%
User Training	7 days	Wed 04/26/17	Mon 05/01/17	100%
Database User Testing – Week of functionality Tests	7 days	Tue 05/02/17	Tue 05/08/17	100%
Database Rollout	1 day	Wed 05/09/17	Wed 05/09/17	100%
ElCaro2.0 Team Meeting – Celebrate Success – Discuss Lessons Learned	1 day	Fri 05/11/17	Fri 05/11/17	100%

3.1 - Project Schedule



Deliverables

a. Sample Entity Relationship Diagram. A more comprehensive ERD is included in Appendix



4.1 - Sample Entity Relationships

**b. TEA Sample Invoice**

A sample invoice is included below. The script used to generate this is included in the appendix. This script successfully checks for all bookings under one itinerary number with PLSQL loops. Formatting has been set with RPAD functionality to ensure the spacing remains the same no matter which client information is pulled.

- - Travel Experts Travel Agency - .

To: Freedon Annunziato
45 Adelphi St. #2W, NE
Calgary, AB
T2V 2K7

Bill Date: 26-JAN-14
Consultant: Jane
Customer No: 206
Passengers:
Itinerary No: 705

Prepared For:
Freedon Annunziato

Trip Details:
01 March 2014
04 March 2014
Asia

Supplier	Description	Booking No.	Start Date	End Date	Price
UNITED AIRLINES ATLANTIC BLUE CROSS	Airlines Travel Insurance	DFWSE2 KKJH243	10-JAN-14 01-MAR-14	15-JAN-14 04-MAR-14	\$900.00 \$250.00

For your convenience the following rewards were applied

Freedon Annunziato Aero Plan 123546889

Trip Total
Sub Total: \$1,150.00
Booking Fee: \$25.00
Total: \$1,175.00

Credit Card used for Purchase: 87590432908754

4.2 Sample Invoice



TEA Sample Commission Reports

A sample commission report is included below. The script used to generate this is included in the appendix. This script successfully checks for the CANCELLED, PAID, PENDING and OVERDUE commission statuses. Formatting has been set with RPAD functionality to ensure the spacing remains the same no matter which client information is pulled

Travel Experts Travel Agency Commission Status Report

Commission Report - CANCELLED Commissions

Company Name	Commission Due Date	Original Prod End Date	Commission Amount	
Air Canada	May 15 2015	Jul 14 2015	\$60.00	CANCELLED
Celebrity Cruises	Sep 20 2014	Nov 19 2014	\$71.00	CANCELLED
United Airlines	Feb 14 2014	Apr 15 2014	\$33.00	CANCELLED
United Airlines	Dec 03 2014	Feb 01 2015	\$41.00	CANCELLED
United Airlines	Aug 29 2014	Oct 28 2014	\$38.00	CANCELLED

4.3 Cancelled Commission Report 1

Travel Experts Travel Agency Commission Status Report

Commission Report - Paid Commissions

Company Name	Commission Due Date	Original Prod End Date	Commission Amount	
Air Canada	May 13 2015	Jul 12 2015	\$35.00	PAID
Celebrity Cruises	May 31 2014	Jul 30 2014	\$76.00	PAID
National Car Rental	Apr 16 2014	Jun 15 2014	\$28.00	PAID
Royal Caribbean International	Jun 14 2014	Aug 13 2014	\$257.00	PAID
Saskatchewan Blue Cross	Apr 17 2014	Jun 16 2014	\$6.00	PAID
United Airlines	Sep 27 2014	Nov 26 2014	\$48.00	PAID
United Airlines	Jan 09 2015	Mar 10 2015	\$135.00	PAID
Western Vacations	Apr 15 2014	Jun 14 2014	\$179.00	PAID

4.4 Paid Commission Report

Travel Experts Travel Agency Commission Status Report

Commission Report - PENDING Commissions

Company Name	Commission Due Date	Original Prod End Date	Commission Amount	
Aces Aerolineas Centrales De Colom	Jun 21 2014	Aug 20 2014	\$70.00	OVERDUE
Air Canada	Jul 29 2016	Sep 27 2016	\$25.00	OVERDUE
Air Canada	Jul 28 2013	Sep 26 2013	\$32.00	OVERDUE
Alamo Rent A Car	Oct 08 2013	Dec 07 2013	\$6.00	OVERDUE
Celebrity Cruises	Jun 13 2014	Aug 12 2014	\$251.00	OVERDUE
Compagnia Italiana Turismo Inc	Jun 24 2014	Aug 23 2014	\$169.00	OVERDUE
Eldertreks	Jun 22 2014	Aug 21 2014	\$210.00	OVERDUE

4.4 Paid Commission Report



Travel Experts Travel Agency Commission Status Report

Commission Report - PENDING Commissions

Company Name	Commission Due Date	Original Prod End Date	Commission Amount	
Air Bc	Jun 09 2016	Aug 08 2016	\$10.00	PENDING
Air Canada	Feb 28 2013	Apr 29 2013	\$60.00	PENDING
Air Canada	Feb 22 2013	Apr 23 2013	\$60.00	PENDING
Alitours Car Rental By Hertz	Oct 08 2013	Dec 07 2013	\$5.00	PENDING
Chateaux & Hotels De France	Sep 27 2014	Nov 26 2014	\$62.00	PENDING
Embassy Suites - Toronto/Markham	Oct 08 2013	Dec 07 2013	\$8.00	PENDING
Outrigger Hotels & Resorts	Apr 16 2014	Jun 15 2014	\$10.00	PENDING
United Airlines	Jan 26 2014	Mar 27 2014	\$38.00	PENDING
United Airlines	Sep 01 2014	Oct 31 2014	\$38.00	PENDING
United Airlines	Sep 29 2014	Nov 28 2014	\$75.00	PENDING
Universe Assistance	Apr 02 2014	Jun 01 2014	\$4.00	PENDING

4.5 Pending Commission Report



Performance and Tuning Recommendations

Tuning the database is required to keep track of any high wait times to access data for either the Agents or the Apex application running on the server. Impacts of tuning your queries can cut result time from 17 seconds to a fraction of a second by having reliable hardware, properly set initialization parameters, reviewing SQL query code and building performance indexes on tables.

The hardware design which the database runs on is below. The ST3100I exceeds the recommended specifications for the database. Below is a quick reference to hardware specifications. These will be further discussed in the *System and Database Analysis* section of this documentation.

Component	Specs	Notes
ME PC V_ST3100I Server System	32 GB RAM I5 E3-1220 processor - 4 threads - quadcore 4x Gigabit Ethernet LAN Ports 5x 1TB SSD HDD	- Upgraded amount of memory will keep CPU from having to keep reading from disk. - Fast quad thread, quad core processor. Fast reads when reading from disk. - Performance up!

4.3 - Hardware Components

Though a copy of Windows 2012R2 server is included with our hardware purchases, the database server will run a Red Hat Linux distribution. Linux has a reputation of being very reliable. Yearlong server uptime (or more!) and the ability to control specifically which services run on the server will keep any external interference to the database to a minimum.

Depending on the version and licensing of Oracle Software, some performance monitoring and advisor tools may not be available. This is the case with the Enterprise version of the Oracle software. In our negotiations with Oracle sales they have agreed to let us use Enterprise for the next 5 years at a significantly discounted rate then allow the option to downgrade if needed.

The following is considered regarding database design

- 3rd normal form - Data is not repeated and is not taking up additional database space because it can be referenced from its own database entity.
- Application code – Having standards for the SQL written for invoice, commission or marketing reports will be standardized to keep code in memory. This is important as any minute changes in code require the CPU to reference the disk again, not taking advantage of the code already in memory. Coding standards will also be applied to all stored procedures and functions.
- Following the best practice of running test code in a dev/test environments avoiding any rogue impact to production performance.
- Before rolling out new hardware – ‘simulate’ production workloads in test environment with Database Replay
- Indexes on primary and foreign keys. Create performance indexes on any frequently queried columns or every column in the SELECT statement (index only execution plan) if the statement is run often to avoid full table scans

Keeping an even balance between performance and budget, without going overboard on either, was our number one priority on the plan above.



Instance Tuning

Instance Tuning refers to setting appropriate initializing parameters for the database to run. Goals are set and database parameters are configured to those specified goals. Sticking to the goal will keep you from over tuning and possibly ruining the performance gains you have already found.

Parameters which are currently set in the database are found by either running the SQL below or going through Enterprise Manager > Server > Database Configuration > Initialization Parameters.

The instance can also benefit from a Resource Plan. Consumer Groups are assigned Resource Plans to control the CPU usage, parallelism, number of active sessions and idle time. Different teams or departments within the Travel Agency can have different priorities to the server processing. To set a resource plan use

```
ALTER SYSTEM resource_manager_plan = plan_name;
```

Invalid database objects will cause code failure and should be monitored. These invalid objects will hamper database performance as well.

Monitor any corrupt database objects by querying

```
SELECT object_name, object_type  
FROM DBA_OBJECTS  
WHERE status = 'INVALID';
```

To rebuild an invalid Index use the commands below. You can also rebuild indexes in Enterprise Manager under Objects > Table. Rebuilding indexes can also be performed on valid DBA_OBJECTS to include any changed data in the table and is recommended to perform weekly (or nightly if busier) to keep table scans minimal, which again tunes the instance.

```
ALTER INDEX indexname REBUILD;
```

Database Patching

Oracle releases patches with new functionality and bug fixes regularly. CPU (critical patch updates) include security fixes and regression tests. The updates will be installed to keep the database running optimally using a tool called OPatch. Before applying a patch to the database, OPatch must be patched to the equivalent version as well.

In the event a patch is required and the database cannot be shut down an Online Patch will be done. Online patching keeps the database running, the process is quick and takes little memory.

Verify if a particular patch release can be run online with the syntax below

```
$ cd $ORACLE_HOME/OPatch  
$ OPatch query -is_online_patch patch location  
$ OPatch query patch location -all
```

Memory Management

Further instance tuning is performed by setting memory management parameters.

Automatic Memory Management (AMM) - Sets the total amount of memory allocated to the database and automatically allocates memory back and forth from the PGA and SGA. If a DBA were to execute the same tasks as the AMM they would need to set the SGA_TARGET (memory ceiling) and the PGA_AGGREGATE_TARGET (20% of the size of the database by default).



Additionally Automatic Shared Memory Management automatically manages memory allocation of all of the SGA (Shared Pool, Java Pool, Buffer Cache, Streams Pool and Large Pool) components. ASMM is automatically enabled under AMM.

Code to determine recommended SGA_TARGET

```
SELECT (SELECT SUM(value) FROM v$sga -  
       (SELECT current_size FROM v$sga_dynamic_free_memory)) "SGA Target"  
      FROM dual;
```

ASMM and AMM also provide charted recommendations on how high to set your memory percentages within Enterprise Manager showing the ratio of available memory to performance gains. This way you will not over allocate memory – essentially wasting resources.

The recommendation to Travel Experts Travel Agency is to run AMM as opposed to manually setting the parameters above. Both can be enabled by using Enterprise Manager's Server tab > Memory Advisor.

Statistics

Cumulative statistics (wait time, time model), metrics (statistic rates) and sampled stats (stats by session, session history, stats by SQL, stats by service) can be gathered and reported on. With these reports any performance issues can be analyzed and resolved. Optimizer statistics also keep track of I/O performance.

The AWR (Automatic Workload Repository) can also gather and displays memory statistics to view within Enterprise Manager. The MMOM (Manageability Monitor) background process takes a snapshot of the database stats every hour by default. Statistics can also be gathered with SQLPLUS in the DBMS_STATS.GATHER_SYSTEM_STATS procedure. The SYS.AUX_STATS table can be queried to view these.

It's worth noting that you can use statistics gathered from another database to compare against the database you are tuning. A test and production setup for example.

The Automatic Database Diagnostic Monitor (ADDM) reports top problems showing in the snapshots and reports them through Enterprise Manager.

SQL Tuning

Statistics on top running SQL or top active sessions can be analyzed to determine bottlenecks in the database. SQL tuning can make the application or user queries faster and more efficient. SQL Tuning Advisor can provide the following benefits

- Identifying and tuning top SQL statements using most resources (I/O and memory) or taking the most time
- Analyzing one statement or many (tuning sets) at a time or from historical AWR information. Duplicate SQL can also be detected
- Runs nightly against major high load SQL statements
- Use automatic tuning to view recommendations on SQL code restructure, performance index builds (access path analysis), etc.
- Validates its own statistics against itself with SQL profiling

Access the SQL Tuning Advisor in Enterprise manager by selecting *Advisor Central* and choosing *Advisor Type*.

When a SQL Statement runs, Oracle determines the best execution plan to retrieve the data set. A cost based approach is used to build the execution plan with each part of the query composing of a 'load' on the plan.

Viewing this plan can be done by using the AUTOTRACE command



```
SQL> ALTER SESSION SET SQL_TRACE=true;
SQL> set autotrace on timing on
```

Optimizing Queries

Selecting a single row by primary key or row id would bring a quicker result than a full table scan (SELECT * FROM). Join statements work more efficiently than Subquery statements and therefor have less cost in the cost based optimizer building the execution plan.

Another SQL Tuning point to note is if a Function is being used in a query, you need to build the index on the function as a “Function based Index”.

```
CREATE INDEX cust_upper_idx ON customers (upper(substring (cust_first,1,1)) customer_ref );
```



System and Database Analysis

Database Recommendation -

The recommended database solution for the TEA database is Oracle Enterprise 11g. Some features included in Oracle 11g are

- Data Pump Data Loading – Import data from Excel spreadsheets or flat csv files, Transfer data from one database to another or transfer user data from one schema to another (in the event of an employee no longer being with the company). This feature will also be helpful in setting up TEST or DEV databases to keep experimental code out of the PROD environment.

```
impdp teaadmin/teaadmin directory =oracleparty
dumpfile = teadb.dmp
tables = gregsseahawksstatistics
logfile = statisticsdata.log
```

5.1 - Data Import Syntax Sample

- Recovery Manager (RMAN) – Backup and recovery will be managed through Oracle's RMAN utility. RMAN allows for automatic recovery by keeping track of which archive logs or redo logs need to be applied to a database to fully recover data up to the time of failure.
- Business Intelligence and Data Warehousing – Materialized views can be loaded with real time data for analysis on product sales, busiest times of the year for bookings, and for commission reporting.
- Audit Controls – Changes to system or user tables will be monitored by Oracle. Any changes or even access to critical data tables can be logged. If there are inconsistencies with user access (late night data copied for examples) they will be logged.

```
AUDIT delete, insert, update, select ON sys.aud$ BY ACCESS;
```

5.2 – Auditing the Audit Trail Table

Hardware Recommendation –

ME PC V_ST3100I Server System is our recommendation for your server. With the hardware specifications below, there is absolute confidence the database will run fast and reliably.

ME PC V_ST3100I Server System	32 GB RAM I5 E3-1220 processor - 4 threads - Quad Core 4x Gigabit Ethernet LAN Ports 5x 1TB SSD HDD
-------------------------------	--

5.3 - Hardware Specifications

Operating System Recommendation –

ElCaro2.0 had a choice for which O/S the server should be running. While running Windows Server 2012 would be easier for users to use – we decided to run RedHat Linux for its stability and security instead. Reasoning behind this is because only technically inclined staff or techs should be using the server directly. Agents connecting remotely via the database will not notice any difference in the user experience.

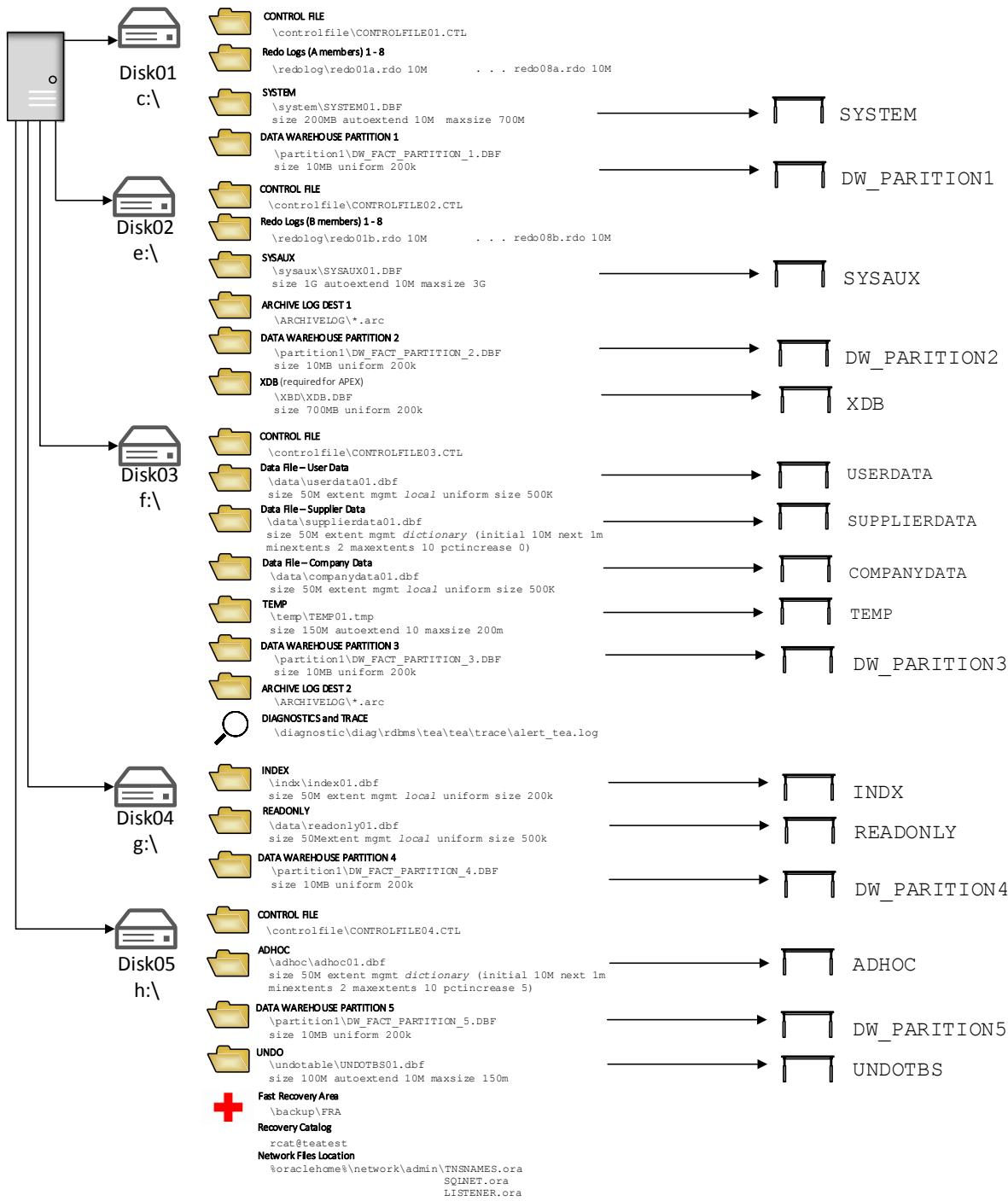


Travel Experts Travel Agency

Database Blueprint
Kenny Vigor
Create Database

Database Files c:\tead\min\createTEA.sql c:\initTEA.ora c:\%oralehome%\database\SPFILEprodacct.ora Log_Mode = ARCHIVELOG Flashback_on = YES	db_domain=TEA db_name=TEA instance_name=TEA compatible=11.2.0.0 db_block_size=16384 db_cache_size=49152 db_files = 500	CHARACTER SET UTF8 max_dump_file_size=10 maxlogfiles 32 maxlogmembers 4 maxdatafiles 75 maxloghistory 3
--	--	--

Datafile
 TableSpace Mapping





Next Steps

Web Integration –

Going forward TEA can benefit by integrating a web application to their database. This will allow agents to pull commission reports or insert customer information without the need to learn SQL code or access any terminal interface.

The front end GUI will allow for data validations and a standard entry form allow agents to enter information for each client and with complete accuracy. Management can use reports, visible only to themselves with customized group assignments, to monitor their business.

Public pages will be available for customers to view new travel package specials and TEA's contact information displayed for easy communication.

The benefits of having data quality validations, such as “The customer’s last name cannot be empty” will ensure records are kept accurate. The ability to enter new customer information by entry form or to check if a customer already exists will give efficient record management to agents without needing to know any ‘back end’ SQL code.

This front end system vastly improves upon the previous method of tracking data in Excel spreadsheets by giving a central management system to all employees. If one agent makes a change, other agents will see this change in real-time.



Welcome to Travel Experts Travel Agency. Let us book your next vacation!

<input type="text"/> Search: Package Startdate <input type="button" value="Go"/> Actions ▾							
	Package Code	Package Name	Package Description	Package Startdate	Package Enddate	Cost	Special Cost
	1	Rocky Mountians Rail	10 day train excursion through the Alberta Rockies	06-MAY-2017	16-MAY-2017	\$2,000.00	\$1,500.00
	2	East Coast Fishing Camp	Fishing 'east coast' style. Fishing and Camping	20-MAY-2017	31-MAY-2017	\$1,500.00	\$1,199.00
	3	Egypt Pyramid	See King Tut's Tomb!! 10 days in Egypt	01-MAY-2017	10-MAY-2017	\$7,000.00	\$6,050.00
	4	Sandy beaches of Mexico	14 days in Mexico, all inclusive	01-MAY-2017	14-MAY-2017	\$2,000.00	\$1,500.00

6.1 Travel Packages



Welcome to Travel Experts Travel Agency. Let us book your next vacation!

This page is for Employees and Agents of Travel Experts Travel Agency to manage customer details.

New customers can also be added by using the CREATE button on the bottom right.

Customer ID	First Name	Last Name	Preferred Agent
	Hiedi	Lopez	JM
	Mardig	Abdou	JD
	Ralph	Alexander	DR
	Sean	Pineda	JM
	Julita	Lippen	JD
	Pierre	Radicola	JL
	Daniel	Wicelinski	JM
	Gary	Aung	JM
	Jeff	Runyan	BD
	Omaira	Grant	DR

6.2 Customer Report and Creation

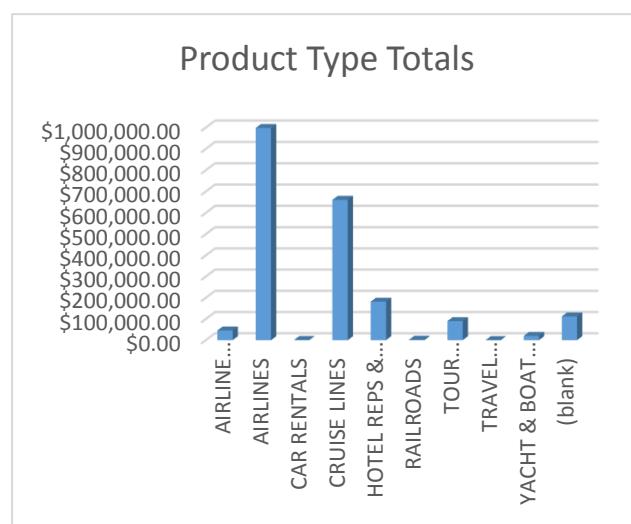
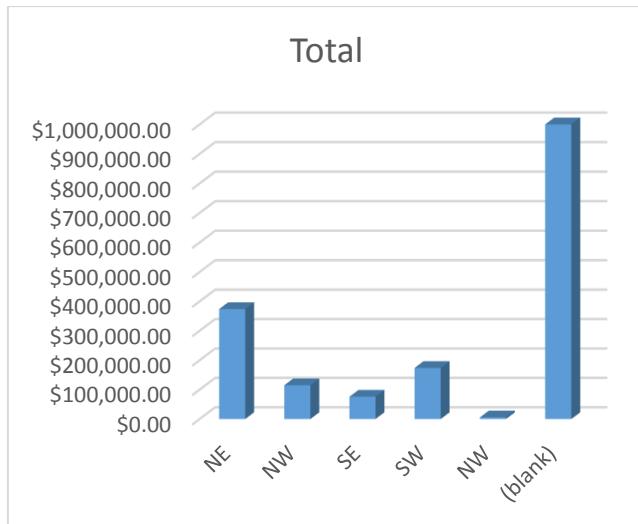


Data Warehouse and Data Mining –

Previously, there was no solution to keep track of the marketing strategies or communications to previous or potential customers (see Appendix 2 Table Instance Charts for CUSTOMERCOMM table). The ability to also track sales by product or sales by type or sales by city quadrant will give more marketing benefits to TEA.

Data collected from the database can also show ‘holes’ in data. Missing or *blank* records (shown below) can be updated to improve data quality on all reports.

Customer Region	Sales by Region	Product Category	Total Sales
NE	\$374,064.00	AIRLINE CONSOLIDATORS	\$45,389.00
NW	\$114,794.00	AIRLINES	\$997,732.00
SE	\$76,091.00	CAR RENTALS	\$800.00
SW	\$173,902.00	CRUISE LINES	\$660,708.00
NW	\$5,600.00	HOTEL REPS & CHAINS	\$181,234.00
(blank)	\$1,365,286.00	RAILROADS	\$1,850.00
Grand Total	\$2,109,737.00	TOUR OPERATORS/WHOLES	\$89,800.00
		TRAVEL INSURANCE	\$315.00
		YACHT & BOAT CHARTERS	\$19,970.00
		(blank)	\$111,939.00
		Grand Total	\$2,109,737.00



6.3 Collection of Data Warehouse Report Samples



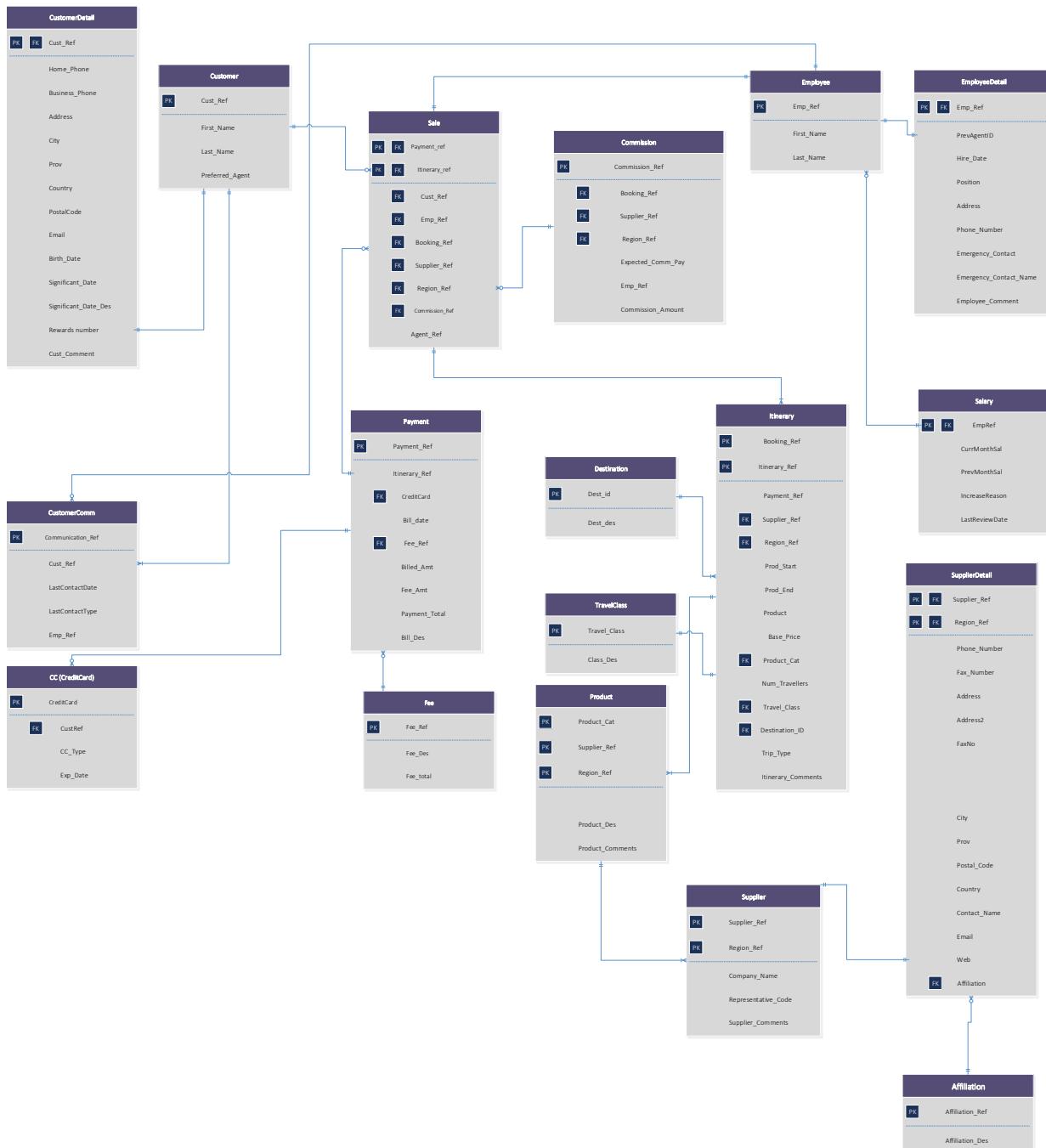
Appendix 1 – Project Schedule

Task Name	Duration	Start	Finish	% Complete
Entire Project Timeline	73 days	Tue 10/25/16	Thu 2/2/17	100%
Analysis	7 days	Tue 10/25/16	Wed 11/2/16	100%
Select Team and Project Mgr	1 day	Tue 10/25/16	Tue 10/25/16	100%
Define the Project Plan (Informational Meetings)	1 day	Wed 10/26/16	Wed 10/26/16	100%
Define Business Requirements	1 day	Thu 10/27/16	Thu 10/27/16	100%
Define Budget Plan	1 day	Fri 10/28/16	Fri 10/28/16	100%
Initial Data Cleanup	3 days	Sat 10/29/16	Wed 11/2/16	100%
Procure Hardware	1 day	Thu 11/3/16	Thu 11/3/16	100%
Puchase Server/Monitors/New Laptops	1 day	Thu 11/3/16	Thu 11/3/16	100%
Design Database	11 days	Fri 11/4/16	Fri 11/18/16	100%
Create ERD - Entities, Attributes and Relationships	7 days	Fri 11/4/16	Mon 11/14/16	100%
BiWeekly Meeting with Stakeholders	4 days	Tue 11/15/16	Fri 11/18/16	100%
Normalization	4 days	Tue 11/15/16	Fri 11/18/16	100%
Develop Database	20 days	Mon 11/21/16	Fri 12/16/16	100%
Create Database Blueprint	1 day	Mon 11/21/16	Mon 11/21/16	100%
Create Pfile	2 days	Tue 11/22/16	Wed 11/23/16	100%
Create Database Script	2 days	Thu 11/24/16	Fri 11/25/16	100%
Create Tablespace Script	2 days	Fri 11/25/16	Mon 11/28/16	100%
BiWeekly Meeting with Stakeholders	0 days	Tue 11/29/16	Tue 11/29/16	100%
Create Database Load Data	20 days	Tue 11/29/16	Mon 12/26/16	100%
Further Data Cleanup	3 days	Tue 12/27/16	Thu 12/29/16	100%
Table Creation	4 days	Fri 12/30/16	Wed 1/4/17	100%
Load Data	5 days	Thu 1/5/17	Wed 1/11/17	100%
Test Database Table Relations	5 days	Thu 1/12/17	Wed 1/18/17	100%
Create Users and Privileges	2 days	Wed 1/18/17	Thu 1/19/17	100%
Confirm User Permissions with TEA	1 day	Fri 1/20/17	Fri 1/20/17	100%
Create User Profiles and Test Limits	2 days	Mon 1/23/17	Tue 1/24/17	100%
Create User Roles	2 days	Wed 1/25/17	Thu 1/26/17	100%
Create User Accounts with Expired Pwds and Lock	1 day	Fri 1/27/17	Fri 1/27/17	100%
Setup Network Access to Database	3 days	Fri 1/27/17	Tue 1/31/17	100%
Planning TNS information (gather computer host addresses)	1 day	Tue 1/31/17	Tue 1/31/17	100%
Setup Tracing/Logging	3 days	Tue 1/31/17	Thu 2/2/17	100%
Setup Server LISTENER.ORA	1 day	Thu 2/2/17	Thu 2/2/17	100%
Setup Client machines TNSNAMES.ORA and SQLNET.ORA	1 day	Fri 2/3/17	Fri 2/3/17	100%
Extensive Network Access Testing	2 days	Sat 2/4/17	Tue 2/7/17	100%
Design GUI Interface	10 days	Wed 2/8/17	Tue 2/21/17	100%
Design User Interface for Entering Data (GUI)	10 days	Wed 2/22/17	Tue 3/7/17	100%
BiWeekly Meeting with Stakeholders	0 days	Tue 3/7/17	Tue 3/7/17	100%
Database Testing	3 days	Wed 3/8/17	Fri 3/10/17	100%
Test DB functionality, test reports and queries	3 days	Wed 3/8/17	Fri 3/10/17	100%
Database Security	3 days	Mon 3/13/17	Wed 3/15/17	100%
Create user profiles, roles, and assign permissions	2 days	Mon 3/13/17	Tue 3/14/17	100%
Create scripts for owner/manager to add new users	1 day	Wed 3/15/17	Wed 3/15/17	100%
Develop Front End GUI	7 days	Thu 3/16/17	Fri 3/24/17	100%
Design GUI interface to load new data	6 days	Thu 3/16/17	Thu 3/23/17	100%
Test process of importing data with GUI	1 day	Fri 3/24/17	Fri 3/24/17	100%
BiWeekly Meeting with Stakeholders	0 days	Fri 3/24/17	Fri 3/24/17	100%
Hardware Setup	6 days	Mon 3/27/17	Mon 4/3/17	100%
Deploy new user computers	2 days	Mon 3/27/17	Tue 3/28/17	100%
Deploy database server	2 days	Wed 3/29/17	Thu 3/30/17	100%
Verify current network infrastructure	2 days	Fri 3/31/17	Mon 4/3/17	100%
Load any further data/Backup Data	6 days	Tue 4/4/17	Tue 4/11/17	100%
Load any additional data	2 days	Tue 4/4/17	Wed 4/5/17	100%
Preform initial backup of data	2 days	Thu 4/6/17	Fri 4/7/17	100%
Preform test data restoration	2 days	Mon 4/10/17	Tue 4/11/17	100%
BiWeekly Meeting with Stakeholders	0 days	Tue 4/11/17	Tue 4/11/17	100%
Rollout	1 day	Wed 4/12/17	Wed 4/12/17	100%



A.1 - Project Schedule

Appendix 2 – ERD and Table Instance Charts

Travel Experts Travel Agency
Entity Relationship Diagram

A.2 - ERD

**TABLE NAME: employee**

Basic employee information is stored in `teaadmin.employee`

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
emp_ref	Primary Key	Unique Not Null	employee		NUMBER	5	00030	00031
first_name		Not Null			VARCHAR2	15	Alex	Jeremy
last_name		Not Null			VARCHAR2	15	Bessler	Vigar

TABLE NAME: employeedetail

Details on employees are stored in `teaadmin.employeedetail`

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
emp_ref	Primary Key Foreign Key	Unique Not Null	employee	emp_ref	NUMBER	5	00030	00031
prevagentid					VARCHAR2	2	AD	JD
hire_date					VARCHAR2	15		
position					VARCHAR2	20	Commission Specialist	Sr Agent
address					VARCHAR2	40	123 Employee Address St	123 Employee Address St
phone_number					NUMBER	10		
emergency_contact					VARCHAR2	10	Kerry	John
emergency_contact_phone					NUMBER	10	403123456	4036543210
employee_comments					VARCHAR2	50	Commission specialist – Prefers Wednesdays off.	

`prevagentid` can be a nullable field to store information on employees who do not have the agent title (ie management, owner). This previous agent ID will be used for historical sales reasons. Going forward, employee numbers will be formalized as a 5 digit unique number.

TABLE NAME: salary

Employee salary information is stored in `teaadmin.salary`

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
emp_ref	Primary Key	Unique Not Null	employee	emp_ref	NUMBER	5	00030	00031
lastreviewdate					DATE		10-FEB-2016	12-FEB-2016
currmonthsalary					NUMBER	7,2	1200.00	1500.00
prevmonthsalary					NUMBER	7,2	1000.00	1200.00
increasereason					VARCHAR2	70	Well performing employee	Promotion to senior agent.

**TABLE NAME: customer**

Basic customer information is stored in `teadmin.customer`

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
cust_ref	Primary Key	Unique Not Null		cc sale customercomm	NUMBER	5	1001	1002
first_name		Not Null			VARCHAR2	15	Hideo	Jarrett
last_name		Not Null			VARCHAR2	15	Goyer	Tutty
preferred_agent	Foreign Key		Agent	AgentID	VARCHAR2	5	A100	A101

TABLE NAME: customercomm

Customer communication history is stored in `teadmin.customercomm`. This information is used for future marketing or determining who has already received promotional information.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
cust_ref	Foreign Key	Unique Not Null	customer	cust_ref	NUMBER	5	1001	1002
communication_ref	Primary Key	Unique Not Null			NUMBER	10	0000000001	0000000002
lastcontactdate					DATE		10-FEB-2016	10-MAY-2017
lastcontacttype					VARCHAR2	20	email	phone (cold call)
emp_ref	Foreign Key		employee	emp_ref	NUMBER	5	00030	00031

TABLE NAME: customerdetail

Detailed customer information is stored in `teadmin.customerdetail`

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
cust_ref	Primary Key Foreign Key	Unique Not Null	customer	cust_ref	NUMBER	5	1001	1002
home_phone					NUMBER	12	403870999	5877898015
business_phone					NUMBER	12	403870999	5877898015
address					VARCHAR2	50	1595 Calverton Court	1350 Avenue
city					VARCHAR2	10	Calgary	Calgary
prov					VARCHAR2	2	AB	AB
country					VARCHAR2	10	CANADA	CANADA



postalcode					VARCHAR2	10	T2M18L	TM31M
email					VARCHAR2	25	ratorres@aol.com	smcdonald@aol.com
birth_date					DATE		06-FEB-57	26-FEB-56
significant_date					DATE		01-JUN-80	26-MAR-96
significant_date_des					VARCHAR2	20	Anniversary	Wife's Birthday
rewards_number					NUMBER	15	8887777889	98880008880
cust_comment					VARCHAR2	50	Check clients email on next visit	Check clients phone number on next visit

TABLE NAME: teaadmin.cc

This table is keeps a list of client credit card information. *This table has restricted permissions to employees and will be audited for any access.* Client credit cards are stored for future reference and links to customer with the cust_ref reference number.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
cust_ref			customer	cust_ref	NUMBER	5	1001	1002
creditcard	Primary Key	Unique Not Null			NUMBER	16	888834387984533	1324635416546130
cc_type					VARCHAR2	10	VISA	AMEX
exp_date					DATE		06-JAN-16	19-JUL-16

TABLE NAME: teaadmin.payment

When a customer pays for a trip/itinerary the Payment table is used to store information on billing dates, fees and amounts, total bill amounts, and the credit card used to pay for the trip. The payment type is described as Full, Deposit, Pending, etc. A payment is made up of the many itineraries. Payment is a parent table to Sale.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
payment_ref	Composite Primary Key Foreign Key	Unique Not Null	sale	payment_ref	NUMBER	5	2048	2049
itinerary_ref	Composite Primary Key Foreign Key	Unique Not Null	sale	itinerary_ref	NUMBER	5	51110	6001
creditcard_usetopay	Foreign Key		cc	credit_card	NUMBER	16	888834387984533	1324635416546130
bill_date					DATE		06-FEB-17	26-FEB-16
base_price					NUMBER	10,2	7500.00	799.88
fee_ref					VARCHAR2	5	RS	NC
fee_amt					NUMBER	10,2	50.00	0.00
billed_amt					NUMBER	10,2	500.00	760.58
total					NUMBER	10,2	7900.00	799.88
bill_des					VARCHAR2	50	Deposit	FULL PAYMENT

TABLE NAME: teaadmin.sale

The sale table acts as a bridge table between employees, customers, itinerary information and payments. This is a child table to Payment and references both itinerary and payment for referential integrity. The Sale table is one of our main tables in the database. The following pages go over the 'branches' of the Sale table and how they relate.



Sale keeps information based on the customer to agent sale process. A booking number is provided and is related to the itinerary table. The agent who made the sale and customer is shown for sales reporting details. A Supplier Ref is unique, but the BookingRef could be reused by another supplier. For this reason, a composite primary key will be used between SuppRef and BookingRef.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
payment_ref	Composite Primary Key Foreign Key	Unique Not Null	payment	payment_ref	NUMBER	5	2048	2049
itinerary_ref	Composite Primary Key Foreign Key	Unique Not Null	itinerary	itinerary_ref	NUMBER	5	51110	6001
booking_ref			itinerary	booking_ref	NUMBER	10	T6657D	LJJ138
region_ref			itinerary	region_ref	NUMBER	1	1	2
supplier_ref			itinerary	supplier_ref	NUMBER	5	11469	780
cust_ref	Foreign Key	Not Null	customer	cust_ref	NUMBER	5	200	204
emp_ref	Foreign Key	Not Null	employee	emp_ref	NUMBER	5	00001	00020
agent_ref					VARCHAR2	2	JD	AJ
commission_ref	Foreign Key	Not Null	commission	commission_ref	NUMBER	5	35	45

agent_ref is being kept for historical sales references.

TABLE NAME: teaadmin.commission

Commission information (including estimated commission, a commission reference and a commission amount) is stored in the commissions table. Commission details from this table joined to the sale table will provide daily commission expectation and totals reports

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
commission_ref	Primary Key Foreign Key	Unique Not Null	sale	commission_ref	NUMBER	5	35	60
booking_ref	Foreign Key		sale	booking_ref	NUMBER	10	LJJ138	LJJ138
supplier_ref	Foreign Key		sale	supplier_ref	NUMBER	5		780
region_ref	Foreign Key		sale	region_ref	NUMBER	1		2
expected_comm_pay					DATE		06-FEB-17	26-FEB-16
agent_ref					VARCHAR2	2	JD	AC
commission_amount					NUMBER	4,2	100.50	25.00

TABLE NAME: itinerary

Itinerary stores information on the start and end dates of each product by booking number and itinerary reference. This links to Sale and Travel Class (reference) tables. An Itinerary is used to combine the Payment information in the Payment table.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
itinerary_ref	Composite Primary Key	Unique Not Null			NUMBER	5	577	1128
payment_ref	Composite Primary Key	Unique Not Null			NUMBER	5	2048	2049
Booking_Ref		Not Null			VARCHAR2	5	LJJ138	LJJ138



supplier_ref	Foreign Key	Not Null	product	supplier_ref	NUMBER	5		780
region_ref	Foreign Key	Not Null	product	region_ref	NUMBER	1		2
product_cat	Foreign Key	Not Null	product	product_cat	NUMBER	5	305	600
product					VARCHAR2	30	Airlines	Hotels Reps & Chains
prod_start					DATE		06-MAY-17	20-FEB-16
prod_end					DATE		05-JUN-17	28-FEB-16
destination_id			destination	destination_id	VARCHAR2	3	MED	NA
num_travellers					NUMBER	2	1	2
travel_class	Foreign Key		travelclass	travel_class	VARCHAR2	5	FST	BSN
trip_type					VARCHAR2	15	Business	Leisure
itinerary_comments					VARCHAR2	70	Additional comments on trip	Additional comments on trip

TABLE NAME: teaadmin.product

Each product has multiple suppliers to choose from. Data type information follows,

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
supplier_ref	Composite Primary Key Foreign Key	Unique Not Null	Supplier	supplier_ref	NUMBER	5	8000	780
region_ref	Composite Primary Key Foreign Key	Unique Not Null	Supplier	region_ref	NUMBER	1		2
product_cat	Composite Primary Key	Unique Not Null			DATE		06-FEB-17	26-FEB-16
product_des					VARCHAR2	2	JD	AC

TABLE NAME: teaadmin.supplier

Each product has multiple suppliers to choose from. Data type information follows.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
supplier_ref	Primary Key	Unique Not Null			NUMBER	5	8000	780
region_ref	Primary Key	Unique Not Null			NUMBER	1		2
representitive_code					DATE		06-FEB-17	26-FEB-16
company_name					VARCHAR2	2	JD	AC
supplier_comments					VARCHAR2	150	Comments on the supplier go here	Comments on the supplier go here

**TABLE NAME: supplierdetail**

Detailed supplier information is stored in teaadmin.supplierdetail

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Max imum Length	Sample Data 1	Sample Data 2
supplier_ref	Primary Key	Unique Not Null			NUMBER	5	8000	780
region_ref	Primary Key	Unique Not Null			NUMBER	1		2
phone_number					NUMBER	12	403870999	5877898015
fax_number					NUMBER	12	403870999	5877898015
address					VARCHAR2	35	1595 Calverton Court	1350 Avenue
address2					VARCHAR2	30	1595 Calverton Court	1350 Avenue
city					VARCHAR2	25	Calgary	Calgary
prov					VARCHAR2	2	AB	AB
postalcode					VARCHAR2	7	T2M18L	TM31M
country					VARCHAR2	20	CANADA	CANADA
email					VARCHAR2	25	ratorres@aol.com	smcdonald@aol.com
contact_name					VARCHAR2	25	RA Torres	S McDonald
web					VARCHAR2	40	www.travelproducts.com	www.bookingsspecialist.com
affiliation_code	Foreign Key		Affiliation	affiliation_ref	VARCHAR2	15	PGY	ACTANEW
detail_comment					VARCHAR2	100	Supplier specific comments can go here	Supplier specific comments can go here

**TABLE NAME: affiliation**

Supplier affiliation information is stored in `teaadmin.affiliation`. This table is keeps a list of unique Affiliation information for the Supplier Detail Table. This table is for referential integrity.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
affiliation_ref	Primary Key	Unique Not Null			VARCHAR2	5	NEW	ACTA
affiliation_des					VARCHAR2	1		Association of Canadian Travel Agents

TABLE NAME: destination

The destination reference table is `teaadmin.destination`. This table is for referential integrity.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
dest_id	Primary Key	Unique Not Null			VARCHAR2	5	MED	SP
dest_des					VARCHAR2	40	Mediterranean	South Pacific

TABLE NAME: fee

The fee reference table is `teaadmin.fee`. This table is for referential integrity and contains description information for related `fee_ref` relations.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
fee_ref	Primary Key	Unique Not Null			VARCHAR2	5	RF	CH
fee_des					VARCHAR2	30	Refund	Change
fee_total					NUMBER	5,2	25	25



Appendix 3 – Invoice Script

```
-- Invoice Creation Script
-- Version 3
-- Kenny Vigar
-- Used for generating invoice reports based on Itinerary Number

SET SERVEROUTPUT ON

DECLARE
    v_itinerary_import number(5):= &itinerary;
    v_cust_f_name customer.first_name%type;
    v_cust_l_name customer.last_name%type;
    v_cust_address customerdetail.address%type;
    v_cust_city customerdetail.city%type;
    v_cust_prov customerdetail.prov%type;
    v_cust_postalcode customerdetail.postalcode%type;
    v_bill_date payment.bill_date%type;
    v_bill_des payment.bill_des%type;
    v_agent_name employee.first_name%type;
    v_cust_ref customer.cust_ref%type;
    v_num_pass itinerary.num_travellers%type;
    v_itinerary_ref itinerary.itinerary_ref%type;
    v_inin_start itinerary.prod_start%type;
    v_inin_end itinerary.prod_end%type;
    v_destination destination.dest_des%type;
    v_prod_des itinerary.product%type;
    v_supplier_name supplier.company_name%type;
    v_base_price itinerary.base_price%type;
    v_booking_ref itinerary.booking_ref%type;
    v_rewards customerdetail.rewards_number%type;
    v_rewards_type customerdetail.rewards_type%type;
    v_creditcard usedtopay payment.creditcard_usedtopay%type;
    v_subtotal number(5);
    v_fees number(5);
    v_total number(5);
    -- cursors

CURSOR c_customer_info IS
    SELECT c.first_name, c.last_name, cd.address, cd.city, cd.prov, cd.postalcode,
    pay.bill_date, pay.bill_des, e.first_name, c.cust_ref, i.num_travellers, i.itinerary_ref,
    cd.rewards_number, cd.rewards_type, pay.creditcard_usedtopay
    FROM customer c join customerdetail cd on c.cust_ref=cd.cust_ref
        join sale sa on c.cust_ref = sa.cust_ref
            join payment pay on sa.payment_ref = pay.payment_ref
            join employee e on sa.agent_ref = e.agent_id
            join itinerary i on sa.booking_ref = i.booking_ref and
    i.itinerary_ref = sa.itinerary_ref
    WHERE i.itinerary_ref = v_itinerary_import;

CURSOR c_trip_details IS
    SELECT i.prod_start, i.prod_end, d.dest_des
    FROM itinerary i join destination d on i.destination_id = d.dest_id
    WHERE i.itinerary_ref = v_itinerary_import;

CURSOR c_booking_info IS
    SELECT s.company_name, i.product, i.booking_ref, i.prod_start, i.prod_end, base_price
    FROM supplier s join product p on s.supplier_ref = p.supplier_ref and s.region_ref =
    p.region_ref
        join itinerary i on i.product_cat = p.product_cat and i.region_ref =
    p.region_ref and p.supplier_ref = i.supplier_ref
    WHERE i.itinerary_ref = v_itinerary_import ;

BEGIN
    OPEN c_customer_info;
```



```

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('******');
DBMS_OUTPUT.PUT_LINE('*****');
DBMS_OUTPUT.PUT_LINE(chr(10)||RPAD(' ',25)||'. - Travel Experts Travel Agency - .');

LOOP
  FETCH c_customer_info INTO
    v_cust_f_name, v_cust_l_name,v_cust_address, v_cust_city, v_cust_prov,
    v_cust_postalcode, v_bill_date,v_bill_des, v_agent_name,
    v_cust_ref, v_num_pass, v_itinerary_ref, v_rewards, v_rewards_type,
    v_creditcard_usdtopay;
  EXIT WHEN c_customer_info%NOTFOUND;
END LOOP;
  close c_customer_info;

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('To: ',48)||' ||chr(9)||'Bill Date: '||v_bill_date);
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_f_name||' ||v_cust_l_name ,48)||'Consultant:
'||v_agent_name);
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_address,48)||'Customer No: '||v_cust_ref);
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_city||',||v_cust_prov ,48)||'Passengers:
'||v_num_pass);
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_postalcode ,48)||'Itinerary No: '||v_itinerary_ref);
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('Prepared For: '); -- need a new cursor for prepared for - with loop if
using traveller table
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_f_name||' ||v_cust_l_name ,48));
DBMS_OUTPUT.PUT_LINE(chr(10));

  OPEN c_trip_details;
LOOP
  FETCH c_trip_details INTO
    v_itin_start, v_itin_end, v_destination;

  EXIT WHEN c_trip_details%NOTFOUND;
END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('Trip Details: ');
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(to_char(v_itin_start, 'DD Month YYYY'), 48));
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(to_char(v_itin_end, 'DD Month YYYY'), 48));
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_destination, 48));
DBMS_OUTPUT.PUT_LINE(chr(10));

  CLOSE c_trip_details;

OPEN c_booking_info;

DBMS_OUTPUT.PUT_LINE(RPAD('Supplier', 20)||RPAD('Description',20)||RPAD('Booking
No.',20)||RPAD('Start Date',20)||RPAD('End Date',20)||RPAD('Price',20));
DBMS_OUTPUT.PUT_LINE(chr(10));
LOOP
  FETCH c_booking_info INTO
    v_supplier_name, v_prod_des, v_booking_ref, v_itin_start, v_itin_end, v_base_price;

  EXIT WHEN c_booking_info%NOTFOUND;

DBMS_OUTPUT.PUT_LINE(RPAD(v_supplier_name ,20)||RPAD(v_prod_des,20)||RPAD(v_booking_ref,
20)||RPAD(v_itin_start, 20)||RPAD(v_itin_end, 20)||RPAD((to_char(v_base_price,
'$99,999.99')),20));

END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

```



```
DBMS_OUTPUT.PUT_LINE(chr(10)||'For your convenience the following rewards were applied');
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10)||chr(9)||v_cust_f_name||' '||v_cust_l_name);
DBMS_OUTPUT.PUT_LINE(chr(9)||chr(9)||chr(9)||chr(9)||RPAD(v_rewards_type, 10)||v_rewards);

CLOSE c_booking_info;

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('Trip Total');

SELECT SUM(i.base_price) subtotal, SUM(p.fee_amt) fees, SUM(i.base_price) + SUM(p.fee_amt)
total into v subtotal, v fees, v total
FROM itinerary i JOIN sale s ON i.booking_ref = s.booking_ref and i.itinerary_ref =
s.itinerary_ref
JOIN payment p ON s.payment_ref = p.payment_ref
WHERE i.itinerary_ref = v_itinerary_import;

DBMS_OUTPUT.PUT_LINE(chr(9)||'Sub Total: ' ||LPAD((to_char(v_subtotal, '$99,999.99')),20));
DBMS_OUTPUT.PUT_LINE(chr(9)||'Booking Fee: ' ||LPAD((to_char(v fees, '$99,999.99')),18));
DBMS_OUTPUT.PUT_LINE(chr(9)||'Total: '||LPAD((to_char(v_total, '$99,999.99')),24));

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(9)||'Credit Card used for Purchase: ' || v_creditcard_usetopay);

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('*****');
END;
/
```

A.3 - *Invoice Script*



```
-- Commission Reporting
-- Version 1

-- Kenny Vigar
-- Used for generating Commission Reports

col company_name format a30
col product_des format a30

SET SERVEROUTPUT ON

-- Declare variables and cursors and exception

DECLARE
v_commission_status varchar2(10) := UPPER(TO_CHAR('&commission_status'));
v_company_name supplier.company_name%type;
v_supplier_ref supplier.supplier_ref%type;
v_region_ref supplier.region_ref%type;
v_product_des product.product_des%type;
v_itinerary_ref itinerary.itinerary_ref%type;
v_booking_ref itinerary.booking_ref%type;
v_itin_end itinerary.prod_end%type;
v_comm_amt sale.commission_amt%type;
v_comm_status sale.commission_status%type;
v_comm_due_date date;

err_comm_status EXCEPTION;

CURSOR c_paid_commission IS
SELECT s.company_name, s.supplier_ref, s.region_ref, p.product_des,
i.itinerary_ref,i.booking_ref, i.prod_end,i.prod_end+60, sa.commission_amt, sa.commission_status
FROM supplier s JOIN product p on s.supplier_ref = p.supplier_ref and s.region_ref = p.region_ref
JOIN itinerary i on p.supplier_ref = i.supplier_ref and p.region_ref = i.region_ref and i.product_cat =
p.product_cat
JOIN sale sa on i.booking_ref = sa.booking_ref and i.itinerary_ref =
sa.itinerary_ref
WHERE sa.commission_status = 'PAID';

CURSOR c_pending_commission IS
SELECT s.company_name, s.supplier_ref, s.region_ref, p.product_des,
i.itinerary_ref,i.booking_ref, i.prod_end,i.prod_end+60, sa.commission_amt, sa.commission_status
FROM supplier s JOIN product p on s.supplier_ref = p.supplier_ref and s.region_ref = p.region_ref
JOIN itinerary i on p.supplier_ref = i.supplier_ref and p.region_ref = i.region_ref and i.product_cat =
p.product_cat
JOIN sale sa on i.booking_ref = sa.booking_ref and i.itinerary_ref =
sa.itinerary_ref
WHERE sa.commission_status = 'PENDING';
```



```
CURSOR c_cancelled_commission IS
SELECT s.company_name, s.supplier_ref, s.region_ref, p.product_des,
i.itinerary_ref,i.booking_ref, i.prod_end,i.prod_end+60, sa.commission_amt, sa.commission_status
FROM supplier s JOIN product p on s.supplier_ref = p.supplier_ref and s.region_ref = p.region_ref
JOIN itinerary i on p.supplier_ref = i.supplier_ref and p.region_ref = i.region_ref and i.product_cat =
p.product_cat
JOIN sale sa on i.booking_ref = sa.booking_ref and i.itinerary_ref =
sa.itinerary_ref
WHERE sa.commission_status = 'CANCELLED';

CURSOR c_overdue_commission IS
SELECT s.company_name, s.supplier_ref, s.region_ref, p.product_des,
i.itinerary_ref,i.booking_ref, i.prod_end,i.prod_end+60, sa.commission_amt, sa.commission_status
FROM supplier s JOIN product p on s.supplier_ref = p.supplier_ref and s.region_ref = p.region_ref
JOIN itinerary i on p.supplier_ref = i.supplier_ref and p.region_ref = i.region_ref and i.product_cat =
p.product_cat
JOIN sale sa on i.booking_ref = sa.booking_ref and i.itinerary_ref =
sa.itinerary_ref
WHERE sa.commission_status = 'OVERDUE';

-- begin
-- paid

BEGIN

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('Travel Experts Travel Agency Commission Status Report');
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

IF v_commission_status = TO_CHAR('PAID') THEN

    DBMS_OUTPUT.PUT_LINE(' Commission Report - Paid Commissions');
    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('Company Name', 30)||RPAD('Commission Due Date',25)||RPAD('Original Prod
End Date',25)||RPAD('Commission Amount',40));
OPEN c_paid_commission;
LOOP
    FETCH c_paid_commission INTO v_company_name, v_supplier_ref, v_region_ref, v_product_des,
v_itinerary_ref,
    v_booking_ref, v_itin_end, v_comm_due_date, v_comm_amt, v_comm_status ;
    EXIT WHEN c_paid_commission%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE(rpad(initcap(v_company_name), 35)||rpad(to_char(v_itin_end, 'Mon
DD YYYY'), 25)||rpad(to_char(v_comm_due_date, 'Mon DD YYYY'),20)
        ||RPAD((to_char(v_comm_amt, '$99,999.99')),24)||v_comm_status);
END LOOP;
```



```
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

CLOSE c_paid_commission;

ELSIF v_commission_status = TO_CHAR('CANCELLED') THEN
    DBMS_OUTPUT.PUT_LINE(' Commission Report - CANCELLED Commissions');
    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('Company Name', 30) || RPAD('Commission Due Date',25) || RPAD('Original Prod
End Date',25) || RPAD('Commission Amount',40));
OPEN c_cancelled_commission;
LOOP
    FETCH c_cancelled_commission INTO v_company_name, v_supplier_ref, v_region_ref,
v_product_des, v_itinerary_ref,
    v_booking_ref, v_itin_end, v_comm_due_date, v_comm_amt, v_comm_status ;
    EXIT WHEN c_cancelled_commission%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE(rpad(initcap(v_company_name), 35) || rpad(to_char(v_itin_end, 'Mon
DD YYYY'), 25) || rpad(to_char(v_comm_due_date, 'Mon DD YYYY'),20)
        || RPAD((to_char(v_comm_amt, '$99,999.99')),24) || v_comm_status);
END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

CLOSE c_cancelled_commission;

ELSIF v_commission_status = TO_CHAR('PENDING') THEN
    DBMS_OUTPUT.PUT_LINE(' Commission Report - PENDING Commissions');
    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('Company Name', 30) || RPAD('Commission Due Date',25) || RPAD('Original Prod
End Date',25) || RPAD('Commission Amount',40));
OPEN c_pending_commission;
LOOP
    FETCH c_pending_commission INTO v_company_name, v_supplier_ref, v_region_ref, v_product_des,
v_itinerary_ref,
    v_booking_ref, v_itin_end, v_comm_due_date, v_comm_amt, v_comm_status ;
    EXIT WHEN c_pending_commission%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE(rpad(initcap(v_company_name), 35) || rpad(to_char(v_itin_end, 'Mon
DD YYYY'), 25) || rpad(to_char(v_comm_due_date, 'Mon DD YYYY'),20)
        || RPAD((to_char(v_comm_amt, '$99,999.99')),24) || v_comm_status);
END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

CLOSE c_pending_commission;
```



```
ELSIF v_commission_status = TO_CHAR('OVERDUE') THEN
    DBMS_OUTPUT.PUT_LINE(' Commission Report - PENDING Commissions');
    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('Company Name', 30) || RPAD('Commission Due Date',25) || RPAD('Original Prod
End Date',25) || RPAD('Commission Amount',40));
OPEN c_overdue_commission;
LOOP
    FETCH c_overdue_commission INTO v_company_name, v_supplier_ref, v_region_ref, v_product_des,
v_itinerary_ref,
    v_booking_ref, v_itin_end, v_comm_due_date, v_comm_amt, v_comm_status ;
    EXIT WHEN c_overdue_commission%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE(rpad(initcap(v_company_name), 35) || rpad(to_char(v_itin_end, 'Mon
DD YYYY'), 25) || rpad(to_char(v_comm_due_date, 'Mon DD YYYY'),20)
        || RPAD((to_char(v_comm_amt, '$99,999.99')),24) || v_comm_status);
END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

CLOSE c_overdue_commission;

END IF;

EXCEPTION
WHEN err_comm_status THEN
    DBMS_OUTPUT.PUT_LINE('** Error ** Not a Valid Commission Status');

END;
/

```

A.3 Commission Report Script 1



Appendix 4 – Performance and Tuning Execution Plans

Execution Plans will help determine if a query is working optimally within the database. Two sample execution plans are below. The first, a full table scan, shows how the time and ‘cost’ is higher than an Index scan.

```
Elapsed: 00:00:00.17
Execution Plan
-----
Plan hash value: 2844954298

| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time       |
|   0 | SELECT STATEMENT   |           | 271   | 5691  |    4   (0)  | 00:00:01  |
|   1 |  TABLE ACCESS FULL | CUSTOMER | 271   | 5691  |    4   (0)  | 00:00:01  |

-----
Statistics
-----
       609 recursive calls
         0 db block gets
       171 consistent gets
       19 physical reads
         0 redo size
  11125 bytes sent via SQL*Net to client
   721 bytes received via SQL*Net from client
    20 SQL*Net roundtrips to/from client
    30 sorts (memory)
     0 sorts (disk)
   271 rows processed
```

A.4.1 - Full Table Scan Sample

```
Elapsed: 00:00:00.06
Execution Plan
-----
Plan hash value: 2571908341

| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time       |
|   0 | SELECT STATEMENT   |           | 1     | 33   |    3   (0)  | 00:00:01  |
|   1 |  TABLE ACCESS BY INDEX ROWID | SALE    | 1     | 33   |    3   (0)  | 00:00:01  |
| * 2 |   INDEX RANGE SCAN  | SALE_IDX | 1     | 1    |    2   (0)  | 00:00:01  |

-----
Predicate Information (identified by operation id):
-----
 2 - access("PAYMENT_REF"=2339)

Statistics
-----
       0 recursive calls
       0 db block gets
       3 consistent gets
       0 physical reads
       0 redo size
  1145 bytes sent via SQL*Net to client
   523 bytes received via SQL*Net from client
    2 SQL*Net roundtrips to/from client
    0 sorts (memory)
    0 sorts (disk)
    1 rows processed
```

A4.2 - WHERE clause on Index Column



EL CARO 2.0

Requirements Documentation

October 21, 2016

Kenny Vigar

kvigar@gmail.com

In the occurrence of a failure regarding the Travel Experts Travel Agency database the following backup methods will be discussed and implemented.

Document	Page
Project Definition	1
Project Scope	2
Business Requirements	3
Data Requirements	4
Technical Requirements	5
Budget Plan	6
Appendix – Organized Notes/Interview Questions	7

1. Project Definition

Project Start Date: 21/10/2016

Project End Date: 30/04/2017

Project Objectives: To provide and easy to use solution for tracking customer bookings, supplier information, commission reporting, marketing trends, and to consolidate company wide information via database implementation and graphical user interface for data entry. We will stay within the \$500,000 budget and keep the options open to resell this solution to other agencies.

Approach: Initially, we will gather the business rules, and processes for Travel Experts Travel Agency. We will design a database around these business rules. The database will be created, tested, and rolled out as a final product. Please see attached schedule for further details.

Task Phase	Task to accomplish	Result
Analysis	-Select team and project manager -Define the Project Plan (Informational Meetings) -Define Business Needs -Define Functional Requirements -Define Data Requirements -Define Budget Plan -Initial Data Cleanup -Develop basic user training strategy -Develop initial user technical support processes -Determine hardware needs based on analysis phase	-Provide project schedule -Provide business requirements document -Delivery Strategy Document
Design	-Design database ERD -Design front end 'easy to use' GUI -Normalize data -Clean up any existing data anomalies -Design testing strategy for database and GUI integration.	-Have a testing strategy to verify product works after it is created -Show a data model map of the entire project
Creation	-Create database -Create tables and load data -Create GUI interface	-Will have a functional database and GUI to proceed to testing phase -Have a fully usable product to show 'first look' to clients through a user demo
Testing	-Use testing strategy from <i>Design</i> phase to ensure constraints, and functionality are implemented correctly -Train users on system	- Ensure product is complete, and does not have any interference for clients performing daily tasks -Users trained on new system
Rollout	-Go Live -Provide support as per contract until transition to internal support team.	-Solution deployed, users trained -System complete and in use

Scope

We will be providing the following for the TEA database/GUI design project

- Requirement gathering/Determine solution
- Analyze current hardware, ensure it fits the requirements or replace
- Hardware procurement
- Budgeting details
- Database design
- 'Easy to use' graphical interface design
- Data cleanup and data importing to the new database
- Database development
- 'Easy to use' graphical interface development
- Report building (market trends, top sales, commission reports)
- Script development (for users to query the system)
- Security implementation (user accounts, database security)
- Database optimization and tuning
- User training and user technical support (as per contract terms)
- System rollout

Out Scope

This project does not include the following, and will need to be negotiated into the contract terms if required

- User support past the agreed date in the contract
- Web integration of system
- System analysis after initial deployment
- On call or emergency support
- Additional employees beyond the assigned technicians (4 technicians included)
- Additional hardware or training not covered in the scope of the project (advanced SQL training, advanced report or script building)

Please continue to next page for requirements documentation

2. Business Requirements:

We will ensure the following,

- Provide an easy to use graphical interface to enter and view client data.
- Develop detailed reports to track marketing trends, top sales agents.
- Deliver commission payout and owing records, as well as payment types from customers.

Functional Requirements (End-users)

No.	Requirement	Mandatory / Would Like
1	Provide graphical interface for data entry.	M
2	Ability to waive the agency fee as per agent discretion.	M
3	Establish customizable reporting options (monthly, yearly, on demand) to view top sales agents, top sales areas, low sales areas, marketing trends or opportunities and customer information.	M
4	Ability to verify commission tracking. Cheques are sent, generally, 60 days after the product is used, based on the return or end date of the product booking.	M
5	A report to generate the invoice based on the client and booking information.	M
6	Tracking the difference between business, group or leisure travel. As well, the option to track which business users may be interested in leisure travel opportunities.	M
7	Eventually would like a web interface for the system for clients to book online.	W

Please continue to next page for further requirements documentation

Data Requirements:

Information Type	Requirements
Spreadsheet and Supplier DVD/Book.	<p>Consolidate all hardcopy and softcopy information to a single source (database). This will be checked for data anomalies and 'cleaned' before entering into the database.</p> <p>There are currently just over 10,000 individual values of data in the spreadsheet to load to the database (pre-cleanup)</p> <p>Supplier product information is also available every six months but can contain out of date information. The information on products available are in either a book (\$75) or DVD (\$300).</p>
Access and storage	<p>All users will have access to the system, but only the commission specialist will have elevated access to update supplier and cost information.</p> <p>Other users will be able to query data and enter new customer/sales records in the system.</p>
Growth	<p>Owner expects to have a 5 – 10% client growth and the database should be able to handle that.</p>
Information stored	<p>Client name, address and phone numbers. Reward or frequent flyer numbers. Credit card information.</p> <p>Booking number must be unique – Currently it is unique by supplier. Commissions may change values. Each Payment is a new invoice.</p>

Please continue to next page for further requirements documentation

Technical Infrastructure Requirements

No.	Requirement
1	System will need full business hour functionality, with maintenance and upgrades occurring after hours.
2	Additional backup, reliability will be provided by an additional server on site.
3	Database will be backed up nightly as the information stored is changed daily.
4	System will be built on Oracle technologies.
5	All agents will be receiving hardware upgrades in the form of new computers, monitors, and a central server to run the database from.
6	The ability to flag records in the system as needing an update. The flag will notify agents that changes to an account are required when it is pulled up in the system.

Please continue to next page for budget documentation

Budget Plan

Resource Type	Quantity	Product	Cost	Details
Hardware	2	Database server	10,000.00	ME PC V_ST3100I Server System
Hardware	10	Business laptop	599.99	Lenovo G50 80KR0015US
Hardware	10	Docking station	219.00	Targus 4K Universal Docking Station
Hardware	20	Monitors	119.99	22M38D-B 21.5in Full HD
Hardware	1	UPS for server	94.99	Back-UPS ES 550VA
Hardware	Misc.	Keyboards/mice/cables	500.00	Misc hardware for connecting and using computers
		Shipping	1,200.00	
		Total	77,969.80	

Resource Type	Quantity	Product	Cost	Details
Software	1	Oracle Database Standard Edition	20,850.00	Oracle Software and licenses
Software	1	Development software (JAVA)	2,600.00	Software to code front end GUI
Software	60	Supplier product info DVD / 5 years	18,000.00	Supplier information to base commission estimate.
				Total 23,450.00

Resource Type	Name	Hours (total)	Cost	Total Cost
Technician	Kenny	560	70/hour	29,200.00
Technician	Luz	296	70/hour	20,720.00
Technician	Tanya	304	70/hour	21,280.00
Network Specialist	Jolanta	40	100/hour	4000.00
Development specific costs	Kenny	40	150/hour	6000.00

Total	81,200.00
-------	-----------

Please continue to next page for budget documentation

Additional Resources	Supplier	Cost	Total Cost
Offsite Storage for backups	Iron Mountain	3000/month over 5 years	180,000.00
Catering	Starbucks		7,800 .00
High Speed Business Internet	Shaw	3500/month over 5 years	17,500.00
Possible web integration*	Kenny Vigar		100,000
Unexpected Costs	10% of total Budget		50000.00

Total	58,700.00
-------	-----------

*Web integration has been classified as out of scope for this project. It has been included in this estimate for your consideration regarding any cost questions you might have.

Sections Total	241,319.00
Tax	16,892.33
Grand Total	258,211.33

Thank you for your consideration of this proposal, please contact me at any time.

Kenny Vigar
kenny.Vigar@edu.sait.ca

Appendix – Organized Notes and Interview Questions

Travel Experts Travel Agency Interview Questions and Considerations

Interview Questions

How many people will be using the system? Owner? Manager? (Referring to license and hardware costs) -
16 agents, all will use the system, 8 workstations.
Owner and Manager will also appreciate laptops purchased for them. Agents will receive new laptops.

Will different types of agents (Jr, Senior) receive different levels of access? Who will have permissions to change/update/delete data in the system?

Not all agents will have same permissions (different database roles).
Everyone will need some form of access.
Any data changes can be performed by the commission expert (senior agent).

Owner will only need to view reports, no need for access.

Will you be keeping expired data (old credit card numbers, etc)? Is there any existing data that is no longer required and can be ‘cleaned up’ before importing to the database? How long do you want to store the data?

We will need to clean the data first, clean inaccuracies

Data will be stored for 5 years minimum

What customer and sales information do you use for marketing/strategies?

Birthdays, anniversaries and time of year where people travel more often. Only birthday information is currently being tracked.

What information would you like tracked in your reports?

Commission status, top sales by client, region, and time of year.

How are you currently keeping track of commissions? How do you compare to what you are given to a supplier to your own records? Are there different types of commission?

Commission specialist keeps this information in a spreadsheet.
Type of commission status – Outstanding, Paid, Invalid

How often would you like your reports generated?

On demand report pulls are required.

How long can your system be offline for planned maintenance or upgrades?

Not at all. Any work impacting maintenance will need to be after hours.

What is your customer/sales growth rate? Will you anticipate any new columns in your spreadsheets (tables added to database) or any additional report customizations or changes after the initial setup?

Estimated 5 – 10% growth.
No need at this time to add any additional columns aside from requests outlined here.

How data is currently shared? Do you use the same spreadsheet and consolidate information or is it all separate?

All agents have their own methods. Some hard copies, some on spreadsheets. This can cause issues with clients speaking with multiple agents and miscommunication.

How often would you like data copied to backups, for possible restore or recovery? Would you like the backups stored in your office or off site at a secure location?

Nightly backup required, as new information is stored every day.
Safe copy in another location sounds appealing.

What scheduling would you have for training employees?

Training will be the same for all agents.

Elevated training for Sr. Staff (manager, owner, commission specialist).

What is the process to book through a supplier?

Contact supplier through the DVD or Book information (SABER) which has all of the pricing and details for the products – though there are frequent errors and out of date information

Is there any other information in hard copy format we need to import?

No

Regarding the implementation of the system, when would you prefer, time wise, to have this done?

During business hours is ok with notice, but would prefer after hours.

When can we schedule follow up meetings with you?

Face to face meetings every 2 weeks.

Regarding the data quality, who can we speak with to clean this data (same agent ID's, typos)?

Flag errors in database, and they will be corrected as clients return to book further trips as long as the records are flagged.

How will you like to categorize the current data as Business, Leisure, or Group trips?

This will only be tracked going forward, and previous data will not be categorized.

What are the payment types that you would like to track?

Down payments/Installments, Full Payment, Final Payment.

The booking fee will show on only the FIRST invoice.

Commissions – should they be stored as % or \$ amounts?

Will be stored as dollar amount, but can be a blank value as well.

Project Considerations

- Confidential data
- Minor errors in data quality need to be fixed
- We have all of the info – no additional information to add
- Load data, flag data (flag or comments column), clean data quality
- Agents keep track of information in different ways
- Most clients are business, some leisure, want to capture more business clients into leisure travel
- Credit card numbers kept on file, not required though
- Supplier information book available as
 - o hard copy (75\$) or DVD (300\$)
 - o may also be inaccurate – need to be able to edit data
- Reward number – keep on file
- Schedule follow up meetings every 2 weeks
- Budget to include everything (hardware, licensing, donuts, our own project mgmt. fees)
- 1 DBA on staff
- Owner needs reports only
- MGR – full access
- 10 computers
- Cannot have agents leave with only copy of data.

Itinerary information

- 1 trip is 1 or more people
- A single trip can have multiple products attached – (hotels, bikes, canoes, camels in Himalayas, TRAVEL INSURANCE)
- Each product has a start and end time – and might overlap
- Products not paid in advanced
- Commission on products can have a \$ or % - will be tracked as \$ amount – or can be NULL
- TEA expects commission, but if the client does not use the product, no commission paid – but TEA doesn't know they won't be getting the commission
- PRODUCT PRICE is per product – not per person
- SABER booking system is online
- Some products found by phone/email/supplier list (updated every 6 months) or found by TEA themselves

Supplier

- Booking number (currently) is UNIQUE – but only unique by supplier – other suppliers may have a booking number similar (consider using a composite primary key on supplier and booking number?)
- Commissions may change for the same product – cannot track hard coded commissions – but need to track these.
- Commission due date is 60 days after the end date of each product

Payment

- Each payment is a new invoice
- Payment types (deposit, multi deposit, final, full)
- Optional agency Fee (at discretion of the agent to waive) – SHOWS ON FIRST INVOICE ONLY

End.

Table Instance Charts

and

Database Relationship Descriptions

November 4, 2016

Kenny Vigar

kvigar@gmail.com

Relationships below have been summarized to supplement the entity relationship diagram (ERD) which follows.

Table	Page
Employee	1
EmployeeDetail	
Salary	2
Customer	
CustomerComm	3
CustomerDetail	
CreditCard (cc)	4
Payment	
Sale	5
Commission	6
Itinerary	
Product	7
Supplier	8
SupplierDetail	
Affiliation	9
Des	
Fee	10

TABLE NAME: employee

Basic employee information is stored in teaadmin.employee

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
emp_ref	Primary Key	Unique Not Null	employee		NUMBER	5	00030	00031
first_name		Not Null			VARCHAR2	15	Alex	Jeremy
last_name		Not Null			VARCHAR2	15	Bessler	Vigar

TABLE NAME: employeedetail

Details on employees are stored in teaadmin.employeedetail

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
emp_ref	Primary Key Foreign Key	Unique Not Null	employee	emp_ref	NUMBER	5	00030	00031
prevagentid					VARCHAR2	2	AD	JD
hire_date					VARCHAR2	15		
position					VARCHAR2	20	Commission Specialist	Sr Agent
address					VARCHAR2	40	123 Employee Address St	123 Employee Address St
phone_number					NUMBER	10		
emergency_contact					VARCHAR2	10	Kerry	John
emergency_contact_phone					NUMBER	10	403123456	4036543210
employee_comments					VARCHAR2	50	Commission specialist – Prefers Wednesdays off.	

prevagentid can be a nullable field to store information on employees who do not have the agent title (ie management, owner). This previous agent ID will be used for historical sales reasons. Going forward, employee numbers will be formalized as a 5 digit unique number.

TABLE NAME: salary

Employee salary information is stored in teaadmin.salary

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
emp_ref	Primary Key	Unique Not Null	employee	emp_ref	NUMBER	5	00030	00031
lastreviewdate					DATE		10-FEB-2016	12-FEB-2016
currmonthsalary					NUMBER	7,2	1200.00	1500.00
prevmonthsalary					NUMBER	7,2	1000.00	1200.00
increasereason					VARCHAR2	70	Well performing employee	Promotion to senior agent.

TABLE NAME: customer

Basic customer information is stored in teaadmin.customer

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
cust_ref	Primary Key	Unique Not Null		cc sale customercomm	NUMBER	5	1001	1002
first_name		Not Null			VARCHAR2	15	Hideo	Jarrett
last_name		Not Null			VARCHAR2	15	Goyer	Tutty
preferred_agent	Foreign Key		Agent	AgentID	VARCHAR2	5	A100	A101

TABLE NAME: customercomm

Customer communication history is stored in `teaadmin.customercomm`. This information is used for future marketing or determining who has already received promotional information.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
cust_ref	Foreign Key	Unique Not Null	customer	cust_ref	NUMBER	5	1001	1002
communication_ref	Primary Key	Unique Not Null			NUMBER	10	0000000001	0000000002
lastcontactdate					DATE		10-FEB-2016	10-MAY-2017
lastcontacttype					VARCHAR2	20	email	phone (cold call)
emp_ref	Foreign Key		employee	emp_ref	NUMBER	5	00030	00031

TABLE NAME: customerdetail

Detailed customer information is stored in `teaadmin.customerdetail`

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
cust_ref	Primary Key Foreign Key	Unique Not Null	customer	cust_ref	NUMBER	5	1001	1002
home_phone					NUMBER	12	403870999	5877898015
business_phone					NUMBER	12	403870999	5877898015
address					VARCHAR2	50	1595 Calverton Court	1350 Avenue
city					VARCHAR2	10	Calgary	Calgary
prov					VARCHAR2	2	AB	AB
country					VARCHAR2	10	CANADA	CANADA
postalcode					VARCHAR2	10	T2M18L	TM31M

email					VARCHAR2	25	ratorres@aol.com	smcdonald@aol.com
birth_date					DATE		06-FEB-57	26-FEB-56
significant_date					DATE		01-JUN-80	26-MAR-96
significant_date_des					VARCHAR2	20	Anniversary	Wife's Birthday
rewards_number					NUMBER	15	8887777889	98880008880
cust_comment					VARCHAR2	50	Check clients email on next visit	Check clients phone number on next visit

TABLE NAME: teaadmin.cc

This table is keeps a list of client credit card information. *This table has restricted permissions to employees and will be audited for any access.* Client credit cards are stored for future reference and links to customer with the cust_ref reference number.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
cust_ref			customer	cust_ref	NUMBER	5	1001	1002
creditcard	Primary Key	Unique Not Null			NUMBER	16	888834387984533	1324635416546130
cc_type					VARCHAR2	10	VISA	AMEX
exp_date					DATE		06-JAN-16	19-JUL-16

TABLE NAME: teaadmin.payment

When a customer pays for a trip/itinerary the Payment table is used to store information on billing dates, fees and amounts, total bill amounts, and the credit card used to pay for the trip. The payment type is described as Full, Deposit, Pending, etc. A payment is made up of the many itineraries. Payment is a parent table to Sale.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
payment_ref	Composite Primary Key Foreign Key	Unique Not Null	sale	payment_ref	NUMBER	5	2048	2049
itinerary_ref	Composite Primary Key Foreign Key	Unique Not Null	sale	itinerary_ref	NUMBER	5	51110	6001
creditcard_usetopay	Foreign Key		cc	credit_card	NUMBER	16	888834387984533	1324635416546130

bill_date					DATE		06-FEB-17	26-FEB-16
base_price					NUMBER	10,2	7500.00	799.88
fee_ref					VARCHAR2	5	RS	NC
fee_amt					NUMBER	10,2	50.00	0.00
billed_amt					NUMBER	10,2	500.00	760.58
total					NUMBER	10,2	7900.00	799.88
bill_des					VARCHAR2	50	Deposit	FULL PAYMENT

TABLE NAME: teaadmin.sale

The `sale` table acts as a bridge table between employees, customers, itinerary information and payments. This is a child table to Payment and references both itinerary and payment for referential integrity. The Sale table is one of our main tables in the database. The following pages go over the ‘branches’ of the Sale table and how they relate.

Sale keeps information based on the customer to agent sale process. A booking number is provided and is related to the itinerary table. The agent who made the sale and customer is shown for sales reporting details. A Supplier Ref is unique, but the BookingRef could be reused by another supplier. For this reason, a composite primary key will be used between SuppRef and BookingRef.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
payment_ref	Composite Primary Key Foreign Key	Unique Not Null	payment	payment_ref	NUMBER	5	2048	2049
itinerary_ref	Composite Primary Key Foreign Key	Unique Not Null	itinerary	itinerary_ref	NUMBER	5	51110	6001
booking_ref			itinerary	booking_ref	NUMBER	10	T6657D	LJ138
region_ref			itinerary	region_ref	NUMBER	1	1	2
supplier_ref			itinerary	supplier_ref	NUMBER	5	11469	780
cust_ref	Foreign Key	Not Null	customer	cust_ref	NUMBER	5	200	204
emp_ref	Foreign Key	Not Null	employee	emp_ref	NUMBER	5	00001	00020
agent_ref					VARCHAR2	2	JD	AJ
commission_ref	Foreign Key	Not Null	commission	commission_ref	NUMBER	5	35	45

`agent_ref` is being kept for historical sales references.

TABLE NAME: teaadmin.commission

Commission information (including estimated commission, a commission reference and a commission amount) is stored in the commissions table. Commission details from this table joined to the sale table will provide daily commission expectation and totals reports

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
commission_ref	Primary Key Foreign Key	Unique Not Null	sale	commission_ref	NUMBER	5	35	60
booking_ref	Foreign Key		sale	booking_ref	NUMBER	10	LJJ138	LJJ138
supplier_ref	Foreign Key		sale	supplier_ref	NUMBER	5		780
region_ref	Foreign Key		sale	region_ref	NUMBER	1		2
expected_comm_pay					DATE		06-FEB-17	26-FEB-16
agent_ref					VARCHAR2	2	JD	AC
commission_amount					NUMBER	4,2	100.50	25.00

TABLE NAME: itinerary

Itinerary stores information on the start and end dates of each product by booking number and itinerary reference. This links to Sale and Travel Class (reference) tables. An Itinerary is used to combine the Payment information in the Payment table.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
itinerary_ref	Composite Primary Key	Unique Not Null			NUMBER	5	577	1128
payment_ref	Composite Primary Key	Unique Not Null			NUMBER	5	2048	2049

Booking_Ref		Not Null			VARCHAR2	5	LJJ138	LJJ138
supplier_ref	Foreign Key	Not Null	product	supplier_ref	NUMBER	5		780
region_ref	Foreign Key	Not Null	product	region_ref	NUMBER	1		2
product_cat	Foreign Key	Not Null	product	product_cat	NUMBER	5	305	600
product					VARCHAR2	30	Airlines	Hotels Reps & Chains
prod_start					DATE		06-MAY-17	20-FEB-16
prod_end					DATE		05-JUN-17	28-FEB-16
destination_id			destination	destination_id	VARCHAR2	3	MED	NA
num_travellers					NUMBER	2	1	2
travel_class	Foreign Key		travelclass	travel_class	VARCHAR2	5	FST	BSN
trip_type					VARCHAR2	15	Business	Leisure
itinerary_comments					VARCHAR2	70	Additional comments on trip	Additional comments on trip

TABLE NAME: teaadmin.product

Each product has multiple suppliers to choose from. Data type information follows,

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
supplier_ref	Composite Primary Key Foreign Key	Unique Not Null	Supplier	supplier_ref	NUMBER	5	8000	780
region_ref	Composite Primary Key Foreign Key	Unique Not Null	Supplier	region_ref	NUMBER	1		2
product_cat	Composite Primary Key	Unique Not Null			DATE		06-FEB-17	26-FEB-16
product_des					VARCHAR2	2	JD	AC

TABLE NAME: teaadmin.supplier

Each product has multiple suppliers to choose from. Data type information follows.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
supplier_ref	Primary Key	Unique Not Null			NUMBER	5	8000	780
region_ref	Primary Key	Unique Not Null			NUMBER	1		2
representitive_code					DATE		06-FEB-17	26-FEB-16
company_name					VARCHAR2	2	JD	AC
supplier_comments					VARCHAR2	150	Comments on the supplier go here	Comments on the supplier go here

TABLE NAME: supplierdetail

Detailed supplier information is stored in teaadmin.supplierdetail

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
supplier_ref	Primary Key	Unique Not Null			NUMBER	5	8000	780
region_ref	Primary Key	Unique Not Null			NUMBER	1		2
phone_number					NUMBER	12	403870999	5877898015
fax_number					NUMBER	12	403870999	5877898015
address					VARCHAR2	35	1595 Calverton Court	1350 Avenue
address2					VARCHAR2	30	1595 Calverton Court	1350 Avenue
city					VARCHAR2	25	Calgary	Calgary

prov					VARCHAR2	2	AB	AB
postalcode					VARCHAR2	7	T2M18L	TM31M
country					VARCHAR2	20	CANADA	CANADA
email					VARCHAR2	25	ratorres@aol.com	smcdonald@aol.com
contact_name					VARCHAR2	25	RA Torres	S McDonald
web					VARCHAR2	40	www.travelproducts.com	www.bookingspecialist.com
affiliation_code	Foreign Key		Affiliation	affiliation_ref	VARCHAR2	15	PGY	ACTANEW
detail_comments					VARCHAR2	100	Supplier specific comments can go here	Supplier specific comments can go here

Referential Integrity Tables

TABLE NAME: affiliation

Supplier affiliation information is stored in `teaadmin.affiliation`. This table keeps a list of unique Affiliation information for the Supplier Detail Table. This table is for referential integrity.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
affiliation_ref	Primary Key	Unique Not Null			VARCHAR2	5	NEW	ACTA
affiliation_des					VARCHAR2	1		Association of Canadian Travel Agents

TABLE NAME: destination

The destination reference table is `teaadmin.destination`. This table is for referential integrity.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
dest_id	Primary Key	Unique Not Null			VARCHAR2	5	MED	SP
dest_des					VARCHAR2	40	Mediterranean	South Pacific

TABLE NAME: fee

The fee reference table is `teaadmin.fee`. This table is for referential integrity and contains description information for related `fee_ref` relations.

Column Name	Key Type	Nulls/ Unique	FK Ref Table	FK Ref Columns	Data Type	Maximum Length	Sample Data 1	Sample Data 2
fee_ref	Primary Key	Unique Not Null			VARCHAR2	5	RF	CH
fee_des					VARCHAR2	30	Refund	Change
fee_total					NUMBER	5,2	25	25

Database Design Recipe

December 20, 2017

Kenny Vigar

kvigar@gmail.com

The planning strategy for the Travel Experts Agency (TEA) database is listed below. Further details on each phase of the plan will be covered in its relative heading later in this document.

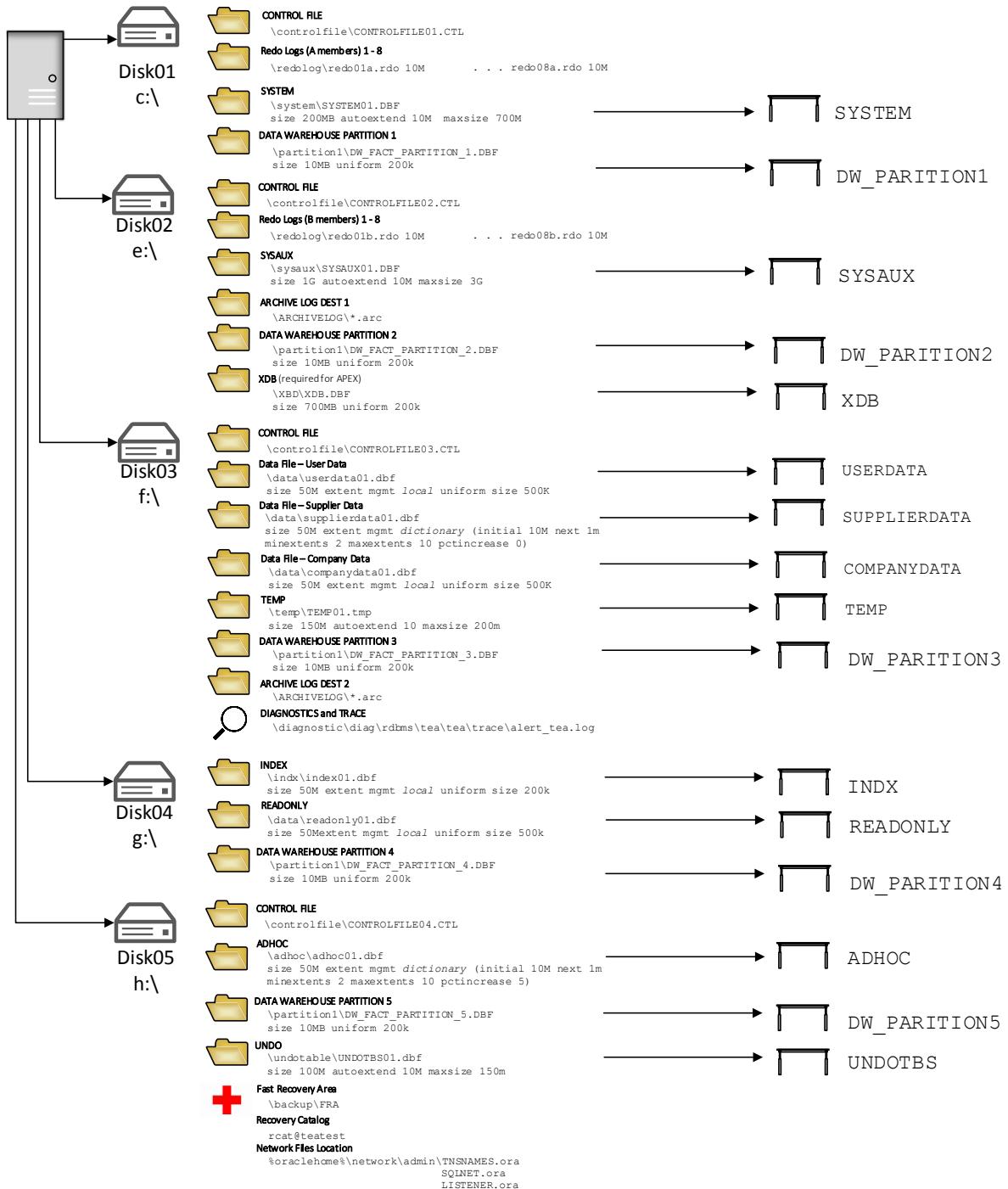
Strategy Stage	Page
Database Blueprint	1
Overview of Environment and Expected Growth - Multiplexing Redo and CTL files	2
Database Creation Recipe (Steps to prepare and create) - Environment Variables - PFile - Control File - Instance - SPFile - Create	2 3
Verification – Are the results what you expected?	4
Create Database Script	5
Create Tablespaces	6
Create DBA User - install catproc, catalog and pupbld	7 8
Appendix 1 - Tablespace Verification Script	9
Appendix 2 - Parameter File	12
Appendix 2 – Entity Relationship Diagram	13

Travel Experts Travel Agency

Database Blueprint
Kenny Vigar
Create Database

Database Files c:\teadmin\createTEA.sql c:\initTEA.ora c:\%oraclehome%\database\\$FILEprodct.ora Log_Mode = ARCHIVELOG Flashback_on = YES	db_domain=TEA db_name=TEA instance_name=TEA compatible=11.2.0.0 db_block_size=16384 db_cache_size=49152 db_files = 500	CHARACTER SET UTF8 max_dump_file_size =10 maxlogfiles 32 maxlogmembers 4 maxdatafiles 75 maxloghistory 3
--	--	---

 Datafile
 TableSpace Mapping



1. Overview

TEA will be storing customer and supplier information in the TEA_KV database. This information will be used more in an analysis or research database as opposed to a heavily transactional OLTP database. For this reason, the design strategy will focus more on a hybrid of an OLTP and DSS (data support system) design.

2. Physical components and growth

All database files will be stored across five discs for redundancy of data. If a disc goes down due to overuse or manufacturer error the other two will recover the In the event third. This configuration also allows for the failure of two discs before any need to recover from an offsite location or cloud storage is needed. Customer data will grow approximately 5% - 10% and the storage available will accommodate this.

According to our Project Management documents, hardware procured have 5x 240GB solid state drives. The datafiles will be multiplexed over the 5 drives.

3. Database install prep - Setting Environment Variables

Because we are borrowing an Oracle Instance, we will only need to verify the oracle_sid
c:\set oracle_sid=ORANT11g

Database install prep – Setup Folder Structure

The create database script will not run unless each of the destination folders listed in the script exist on the disk the database is created on. The attached blueprint shows the layout of the disk structure.

4. Parameter Initialization File

The parameter file loads our instance settings for creating our database. Important parameters loading in the instance are below. A complete pfile is included in the appendix.

```
db_name=TEA
instance_name=orant11g
compatible=11.2.0.0.0
db_block_size=16384
db_cache_size=49152
db_files = 500
sga_max_size = 200M
shared_pool_size = 500M
```

The control files, and diagnostic files are set as

```
diagnostic_dest=C:\TEA\disk03\diagnostic
```

```
control_files=("C:\tea\disk01\controlfile\controlfile01.ctl",
               "C:\tea\disk02\controlfile\controlfile02.ctl",
               "C:\tea\disk03\controlfile\controlfile03.ctl",
```

```
"C:\tea\disk05\controlfile\controlfile04.ctl")
```

*Further initialization parameter settings can be found in the attached initTEA.ora file.

Load the Instance

- 1) Load SQLPLUS
Start > Run > CMD
c:\SQLPLUS /nolog
- 2) Connect to SQL Plus as SysDBA
SQL>conn sys as sysdba
Password: oracle

- 3) "Connected to an idle instance" shows on screen. Load the pfile in NoMount.

```
SQL>startup nomount pfile='c:\TEA\initTEA.ora'
```

BACKGROUND_DUMP_DEST and USER_DUMP_DEST parameter will show as deprecated

(This step is handled automatically in the create database script.)

5. Create the Database

ORACLE instance will start

- 1) Run createTEA.SQL script in NoMount
@c:\TEA\createTEA.sql
Database components created in script.
 - System Tablespace
 - Sysaux Tablespace
 - Undo Tablespace
 - Temp Tablespace
 - Redo Log Files

Additional tablespaces are created after the database creation completes. Tablespaces are

- UserData
- SupplierData
- Index
- ReadOnly
- Adhoc

6. Create spfile from pfile

- 1) In SQL Plus type,

```
CREATE SPFILE = 'c:\tea\disk05\spfileTEA.ora'
FROM PFILE = 'c:\tea\initTEA.ora';
```

7. Verification

- 1) Install data dictionary (catalog)

```
SQL>@C:\app\Administrator\product\11.2.0\dbhome_1\RDBMS\ADMIN\catalog.sql
```

- 2) Install PL/SQL scripts (catproc)

```
SQL>@C:\app\Administrator\product\11.2.0\dbhome_1\RDBMS\ADMIN\catproc.sql
```

- 3) Verify both catalog and catproc ran successfully by querying status, and procedure from dba_registry. This is also included in the attached vdollar.sql verification script.

```
SQL>SELECT status, procedure FROM dba_registry;
```

- 4) Connect as System

```
SQL> conn system/oracle
```

- 5) Run the pupbld.sql script for SQL Plus User Profile

```
SQL>@C:\app\Administrator\product\11.2.0\dbhome_1\sqlplus\admin\pupbld.sql
```

- 6) Verify pupbld ran correctly by logging in as DBA created in step 3. As well, verify the below table exists.

```
SQL> DESC user_product_profile;
```

- 7) Verify tablespaces, instance parameters and datafiles. Attached vdollar.sql will verify these settings.

```
SQL> @c:\tea\vdollar.sql;
```

Once the steps above are complete, we are ready to create tables and load data to the database.

Appendix 1 – Create Database Script

```
#Create Database Script
#Create Tablespace Script

Spool C:\workshop3\createdatabasespool.txt
SET ECHO ON
SET VERIFY ON
set linesize 150

startup nomount pfile='c:\tea\initTEA.ora'

--Verify pfile parameters initialized ok

set head on
COL "Non Default Parameters" format A37
COL "Currently Set as/to" format A60
select name "Non Default Parameters", display_value "Currently Set as/to"
from v$parameter
where isdefault = 'FALSE'
order by name asc;

--Create Database

CREATE DATABASE TEA
USER sys IDENTIFIED BY oracle
USER system IDENTIFIED BY manager
CHARACTER SET UTF8
LOGFILE
GROUP 1 ('C:\TEA\disk01\redolog\redo01a.rdo','c:\TEA\disk02\redolog\redo01b.rdo','c:\TEA\disk03\redolog\redo01c.rdo',
'c:\TEA\disk04\redolog\redo01d.rdo') SIZE 30m,
GROUP 2 ('C:\TEA\disk01\redolog\redo02a.rdo','c:\TEA\disk02\redolog\redo02b.rdo','c:\TEA\disk03\redolog\redo02c.rdo',
'c:\TEA\disk04\redolog\redo02d.rdo') SIZE 30m,
GROUP 3 ('c:\TEA\disk01\redolog\redo03a.rdo','c:\TEA\disk02\redolog\redo03b.rdo','c:\TEA\disk03\redolog\redo03c.rdo',
'c:\TEA\disk04\redolog\redo03d.rdo') SIZE 30m
maxlogfiles 32
maxlogmembers 4
maxdatafiles 75
maxloghistory 3
DATAFILE 'c:\TEA\disk01\system\system01.dbf' SIZE 200m
AUTOEXTEND ON NEXT 10m MAXSIZE 700m, 'c:\TEA\disk04\system\system01.dbf' SIZE 200m
AUTOEXTEND ON NEXT 10m MAXSIZE 700m
SYSAUX DATAFILE 'c:\TEA\disk02\sysaux\sysaux01.dbf' SIZE 120m
AUTOEXTEND ON NEXT 10m MAXSIZE 150m
UNDO TABLESPACE undotbs
DATAFILE 'c:\TEA\disk05\undotable\undotbs01.dbf' SIZE 100m
AUTOEXTEND ON NEXT 10m MAXSIZE 150m
DEFAULT TEMPORARY TABLESPACE temp
TEMPFILE 'c:\TEA\disk03\temp\temp01.tmp' SIZE 150m
AUTOEXTEND ON NEXT 10m MAXSIZE 200m;
```

```
CREATE TABLESPACE userdata
DATAFILE 'C:\TEA\disk03\data\USERDATA01.dbf' SIZE 50M
EXTENT MANAGEMENT local
UNIFORM SIZE 500k;

CREATE TABLESPACE supplierdata
DATAFILE 'C:\TEA\disk03\data\SUPPLIERDATA01.dbf' SIZE 50M
EXTENT MANAGEMENT dictionary
DEFAULT STORAGE (
INITIAL 10M
NEXT 1M
MINEXTENTS 2
MAXEXTENTS 10
PCTINCREASE 0);

CREATE TABLESPACE adhoc
DATAFILE 'C:\TEA\disk05\adhoc\adhoc01.dbf' SIZE 50M
EXTENT MANAGEMENT dictionary
DEFAULT STORAGE (
INITIAL 10M
NEXT 1M
MINEXTENTS 2
MAXEXTENTS 10
PCTINCREASE 5);

CREATE TABLESPACE indx
DATAFILE 'C:\TEA\disk04\indx\index01.dbf' SIZE 50M
EXTENT MANAGEMENT local
UNIFORM SIZE 200k;

CREATE TABLESPACE readonly
DATAFILE 'C:\TEA\disk04\readonly\readonly01.dbf' SIZE 50M
EXTENT MANAGEMENT local
UNIFORM SIZE 500k;

ALTER TABLESPACE readonly READ ONLY;

PAUSE

CREATE SPFILE = 'c:\tea\spfileTEA.ora'
FROM PFILE = 'c:\tea\initTEA.ora';

pause
```

```

--create dba
create user TEADBA
identified by teadba
default tablespace userdata
temporary tablespace temp
quota 10m on readonly
quota 10m on userdata
quota 10m on indx
quota 10m on adhoc;

grant dba to TEADBA;
grant create session to TEADBA;
grant create tablespace to TEADBA;
grant create table to TEADBA;
grant drop any table to TEADBA;
grant drop tablespace to TEADBA;

pause

prompt Install Catalog

SPOOL OFF

--Install data dictionary
@C:\app\Administrator\product\11.2.0\dbhome_1\RDBMS\ADMIN\catalog.sql

Spool C:\workshop3\createdatabasespool.txt append
PROMPT catalog has completed, proceeding to install catproc

PAUSE
SPOOL OFF

--Install PL/SQL scripts
@C:\app\Administrator\product\11.2.0\dbhome_1\RDBMS\ADMIN\catproc.sql

Spool C:\workshop3\createdatabasespool.txt append

-- verify catalog and catproc installed
col status format a8
col procedure format a40
SELECT status, procedure FROM dba_registry;

PROMPT catalog has completed, catproc completed - Installing pupbld

```

```
--conn system/oracle for pupbld
conn system/manager
PAUSE
SPOOL OFF

@C:\app\Administrator\product\11.2.0\dbhome_1\sqlplus\admin\pupbld.sql

Spool C:\workshop3\createdatabasespool.txt append
PROMPT catalog has completed, catproc has completed, pupbld has completed
--Verify Pupbld has completed by checking product_user_profile
DESC product_user_profile;

-- double verify pupbld by logging in as new dba
conn teadba/teadba
```

```

--verify tablespaces

Clear Col

set head off
Prompt ****
PROMPT ** Verification Script **
Prompt ****
SELECT banner from v$version;
show user

SELECT '1 Database Name: '||name FROM v$database
UNION
SELECT '2 Instance Name: '|| instance_name FROM v$instance
UNION
SELECT '3 Host Name: '|| host_name FROM v$instance
UNION
SELECT '4 Todays date : '||sysdate FROM dual;
Prompt ****
PROMPT Press Enter When Ready
Prompt ****
PAUSE
set head off

Prompt ****
PROMPT Verify CTL, DBF and TMP
Prompt ****

SELECT 'Control Files: '||name FROM v$controlfile;

set linesize 150

--set head on
--COL "Table Space Name" format A20
--COL "Size in Bytes" format 99999999999
--COL Location format A45
--SELECT ts.name "Table Space Name", df.name Location, df.bytes Bytes, ((df.bytes/1024)/1024) "MB"
--FROM v$datafile df JOIN v$tablespace ts ON ts.ts# = df.ts#;

Prompt ****
PROMPT Press Enter For Redo Log Info
Prompt ****
PAUSE

col member format a30
set pagesize 50

```

```

set head on

COL GROUP# format 999
COL Status format A12
COL Location format A40
BREAK ON group# ON members ON sequence ON status skip 1
select l.group#, l.members Members, l.sequence# Sequence, l.status "Status", lf.member Location, l.bytes Bytes,
((l.bytes/1024)/1024) "MB"
from v$log l JOIN v$logfile lf on lf.group# = l.group#;

set head off
Prompt *****
PROMPT Press Enter When Ready for Parameter Verification Is Not Default
Prompt *****
PAUSE

PROMPT

set head on
COL "Non Default Parameters" format A37
COL "Currently Set as/to" format A60
select name "Non Default Parameters", display_value "Currently Set as/to"
from v$parameter
where isdefault = 'FALSE'
order by name asc;

col status format a10
col procedure format a45
col schema format a10

select schema, status, procedure, modified from dba_registry;

col username format a10
col default_tablespace format a10
col temporary_tablespace format a6
select username, account_status, default_tablespace, temporary_tablespace from DBA_USERS;

set linesize 165

set head on
set linesize 130

COL "Table Space" format A8 heading "Table|Space"
COL Location format A14 word_wrapped
COL MB format 999
COL "Extent Mgmt" format A10 heading "Extent|Mgmt"
COL min_extents format 999 heading "Min|Extents"
COL max_extents format 999999999 heading "Max|Extents"
COL pct_increase format 99 heading "PCT|INCREASE"

```

```
SELECT ts.name "Table Space", df.name Location, dbats.EXTENT_MANAGEMENT "Extent Mgmt", ((df.bytes/1024)/1024) "MB", dbats.max_size,
dbats.min_extents, dbats.max_extents, dbats.PCT_INCREASE
FROM v$logfile df JOIN v$tablespace ts ON ts.ts# = df.ts# JOIN dba tablespaces dbats on dbats.tablespace_name = ts.name
UNION
SELECT ts.name "Temp File", tf.name Location, dbats.extent_management, ((tf.bytes/1024)/1024) "MB", dbats.max_size, dbats.min_extents,
dbats.max_extents, dbats.PCT_INCREASE
FROM v$tempfile tf JOIN v$tablespace ts ON ts.ts# = tf.ts# JOIN dba tablespaces dbats on dbats.tablespace_name = ts.name
order by "MB" DESC;
--Verify pupbld with describing user profile
desc product_user_profile;

SPOOL OFF
```

Appendix 1 – Travel Experts Database Parameter File

```
#initTEA pfile
##Last modified by - Kenny Vigar

# add listener
local_listener=TRAVELLISTENER

db_name=TEA
instance_name=tea
compatible=11.2.0.0.0
db_block_size=16384
db_cache_size=49152

db_files=500

#sga_max_size=600M
shared_pool_size=600M

#largepool for shared server SGA
large_pool_size=500M

sga_target=400M

#added below for apex install
MEMORY TARGET = 1600M
#parameters above for Automatic Memory Management

diagnostic dest=C:\TEA\disk03\diagnostic
max_dump_file_size=1000

#ctl and temp
control_files=("C:\tea\disk01\controlfile\controlfile01.ctl",
               "C:\tea\disk02\controlfile\controlfile02.ctl",
               "C:\tea\disk03\controlfile\controlfile03.ctl",
               "C:\tea\disk05\controlfile\controlfile04.ctl")

#undo
undo_management=AUTO
undo_tablespace=undotbs
#undo_tablespace=undotbs2
#undo_management=manual
#transactions=40
#transactions_per_rollback_segment=5
#rollback_segments=(rbs01,rbs02,rbs03)
remote_login_passwordfile=exclusive

O7 dictionary accessibility=false
#restrict remote authentication

#user parameters
RESOURCE_LIMIT=TRUE
```

```

#audit information
#audit_trail=os
audit trail=db
AUDIT_SYS_OPERATIONS=true

#deprecated parameters
#max_enabled_roles=148
#remote_os_authent=false
#background_dump_dest=C:\TEA\disk03\diagnostic\background_dump
#user_dump_dest=C:\TEA\disk03\diagnostic\user_dump

DISPATCHERS = "(protocol=tcp) (DISPATCHERS=4)"

#create shared server
max shared servers=12
SHARED_SERVERS=6
MAX_DISPATCHERS=5
CIRCUITS=100

# archive log mode

# for 9i startup
LOG_ARCHIVE_START=TRUE
LOG_ARCHIVE_MAX_PROCESSES=2
LOG_ARCHIVE_DEST_1 = "location=C:\TEA\Disk02\archivelog"
LOG_ARCHIVE_DEST_2 = "location=C:\TEA\Disk03\archivelog"

FAST_START_MTTR_TARGET=600
RECOVERY_PARALLELISM=4
FAST START PARALLEL ROLLBACK=HIGH
#%s is sequence number of the archive log order
#%r reset number
#%t used for rack
LOG_ARCHIVE_FORMAT = %%ORACLE_SID%%T%S%r.ARC

```

Create Tables

Load Data

January 4, 2016

Kenny Vigar

kvigar@gmail.com

Table	Page
Code Clean Up – Prep (drop existing tables of same name)	1
Ad Hoc Table Creation	5
- Supplier	
- Sales	
- Customer	
Data Cleanup with SQL	10
Create Employee Table	25
Create Employee Detail Table	26
Create Salary Table	27
Create Customer Table	32
Create Customer Details Table	33
Create Credit Card Table – Clean up Credit Card Data	34
Create Customer Communications Table	39
Create Fee Table	43
Create Destination Table	44
Create Travel Class Table	47
Create Affiliation Table	49
Create Supplier Table	50
Create Supplier Details Table	52
Create Product Table	56
Create Itinerary – Clean up Itinerary Data	60
Create Payment Table	64
Create Sale Table	66
Create Commission Table	68
Verify all tables/data/datatypes/counts	69
SQLDR Logs and CTL Files	90

```
SQL> SET VERIFY ON
SQL> TTITLE OFF
SQL>
SQL> -- Spool 1
SQL>
SQL>
SQL> -- Control Files and Logs stored at C:\TEA\csv\control\ and C:\TEA\csv\control\controllogs
SQL> -- Copy and paste from command prompt has been added to the spool document manually (as it is run in CMD not, SQLPLUS)
SQL>
SQL>
SQL>
SQL>
SQL> -- TEA CREATE TABLES AND LOAD DATA SCRIPT
SQL> -- LAST REVISED JAN 2017, Kenny Vigar
SQL>
SQL>
SQL> -----
SQL> -- Before running script
SQL> -- - drop all tables, and sequences below
SQL> -- - expecting errors below if tables do not exist
SQL> -- - delete all control log files before running (c:\tea\csv\control\controllogs)
SQL> -----
SQL>
SQL> drop table teaadmin.adhoc_allclient cascade constraints;
Table dropped.

SQL> drop table teaadmin.adhoc_allsales cascade constraints;
Table dropped.

SQL> drop table teaadmin.adhoc_allsupplier cascade constraints;
Table dropped.

SQL> drop table teaadmin.affiliation cascade constraints;
drop table teaadmin.affiliation cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.cc cascade constraints;
drop table teaadmin.cc cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist
```

```
SQL> drop table teaadmin.customer cascade constraints;
drop table teaadmin.customer cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.customercomm cascade constraints;
drop table teaadmin.customercomm cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.customerdetail cascade constraints;
drop table teaadmin.customerdetail cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.employee cascade constraints;
drop table teaadmin.employee cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.employeedetail cascade constraints;
drop table teaadmin.employeedetail cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.itinerary cascade constraints;
drop table teaadmin.itinerary cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.payment cascade constraints;
drop table teaadmin.payment cascade constraints
*
ERROR at line 1:
```

```
ORA-00942: table or view does not exist

SQL> drop table teaadmin.product cascade constraints;
drop table teaadmin.product cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.salary cascade constraints;
drop table teaadmin.salary cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.supplier cascade constraints;
drop table teaadmin.supplier cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.supplierdetail cascade constraints;
drop table teaadmin.supplierdetail cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.travelclass cascade constraints;
drop table teaadmin.travelclass cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.commission cascade constraints;
drop table teaadmin.commission cascade constraints
*
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> drop table teaadmin.destination cascade constraints;
drop table teaadmin.destination cascade constraints
*
```

```
ERROR at line 1:  
ORA-00942: table or view does not exist
```

```
SQL> drop table teaadmin.fee cascade constraints;  
drop table teaadmin.fee cascade constraints  
*  
ERROR at line 1:  
ORA-00942: table or view does not exist
```

```
SQL> drop table teaadmin.producttemp cascade constraints;  
drop table teaadmin.producttemp cascade constraints  
*  
ERROR at line 1:  
ORA-00942: table or view does not exist
```

```
SQL> drop table teaadmin.producttemp2 cascade constraints;  
drop table teaadmin.producttemp2 cascade constraints  
*  
ERROR at line 1:  
ORA-00942: table or view does not exist
```

```
SQL> drop table teaadmin.sale cascade constraints;  
drop table teaadmin.sale cascade constraints  
*  
ERROR at line 1:  
ORA-00942: table or view does not exist
```

```
SQL>  
SQL>  
SQL> drop sequence teaadmin.comm_ref_seq;  
drop sequence teaadmin.comm_ref_seq  
*  
ERROR at line 1:  
ORA-02289: sequence does not exist
```

```
SQL>
SQL> -----
SQL> --
SQL> -- Create Adhoc Tables
SQL> --   - Constraints on the Adhoc tables have been intentionally left out as they are temp tables
SQL> --
SQL> -----
SQL> 
SQL> 
SQL> -----
SQL> -- ADHOC_ALLCLIENT Table
SQL> --   Data dump of TEA provided Sales documentation
SQL> 
SQL> -- TABLESPACE - ADHOC
SQL> -- SOURCE C:\TEA\csv\clientdata.csv
SQL> -- 270 Rows, Skip 1
SQL> -- SOURCE NOTES - customers tab of provided Excel file
SQL> --
SQL> -- After table creation run in c:\ 
SQL> -- SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\adhocclientcontrol.ctl
SQL> -- log=C:\tea\csv\control\controllogs\adhocclientcontrol.log bad=C:\tea\csv\control\controllogs\badadhocclient.log
SQL> -----
SQL> 
SQL> 
SQL> CREATE TABLE teaadmin.adhoc_allclient
  2 (cust_id number(3),
  3  firstname varchar2(20),
  4  lastname varchar2(20),
  5  agent varchar2(2),
  6  email varchar2(25),
  7  homephone number(12),
  8  businessphone number(10),
  9  birthdate DATE,
 10 Address varchar(35),
 11 city varchar2(10),
 12 postalcode varchar2(7),
 13 prov varchar2(2),
 14 country varchar2(10),
 15 comments varchar2(50))
 16 TABLESPACE adhoc
 17 STORAGE (initial 50k);
```

Table created.

```

SQL>
SQL> desc adhoc_allclient;
Name Null? Type
-----
CUST_ID NUMBER(3)
FIRSTNAME VARCHAR2(20)
LASTNAME VARCHAR2(20)
AGENT VARCHAR2(2)
EMAIL VARCHAR2(25)
HOMEPHONE NUMBER(12)
BUSINESSPHONE NUMBER(10)
BIRTHDATE DATE
ADDRESS VARCHAR2(35)
CITY VARCHAR2(10)
POSTALCODE VARCHAR2(7)
PROV VARCHAR2(2)
COUNTRY VARCHAR2(10)
COMMENTS VARCHAR2(50)

SQL>
SQL>
SQL>
SQL>
SQL> -----
SQL> -- ADHOC_ALLSUPPLIER Table
SQL> -- Data dump of TEA provided Sales documentation
SQL> -- TABLESPACE - ADHOC
SQL> -- SOURCE C:\TEA\csv\suppdata.csv
SQL> -- 1193 Rows, Skip 1
SQL> -- SOURCE NOTES - supplier tab of provided Excel file
SQL> --
SQL> -- After table creation run in c:\
SQL> -- SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\adhocsuppliercontrol.ctl
log=C:\tea\csv\control\controllogs\adhocsuppliercontrol.log bad=C:\tea\csv\control\controllogs\badadhocsupplier.log
SQL> -----
SQL>
SQL>
SQL> CREATE TABLE adhoc_allsupplier
 2 (supplier_id number(5),
 3 product_cat number(3),
 4 Supp_office number(1),
 5 Prod_Des varchar2(30),
 6 Contact varchar2(25),
 7 Company varchar2(75),
 8 Supp_Address varchar2(35),
 9 Supp_Address2 varchar2(30),
10 city varchar2(25),
11 prov varchar2(2),
12 postalcode varchar2(7),

```

```

13    country varchar2(20),
14    Phone number(12),
15    Fax number(12),
16    email varchar2(45),
17    web varchar2(40),
18    representative varchar2(35),
19    affiliation varchar2(15),
20    comments varchar2(100))
21  TABLESPACE adhoc
22  STORAGE (initial 50k);

```

Table created.

```

SQL>
SQL> desc adhoc_allclient;

```

Name	Null?	Type
CUST_ID		NUMBER(3)
FIRSTNAME		VARCHAR2(20)
LASTNAME		VARCHAR2(20)
AGENT		VARCHAR2(2)
EMAIL		VARCHAR2(25)
HOMEPHONE		NUMBER(12)
BUSINESSPHONE		NUMBER(10)
BIRTHDATE		DATE
ADDRESS		VARCHAR2(35)
CITY		VARCHAR2(10)
POSTALCODE		VARCHAR2(7)
PROV		VARCHAR2(2)
COUNTRY		VARCHAR2(10)
COMMENTS		VARCHAR2(50)

```

SQL>
SQL> -----
SQL> -- ADHOC_ALLSALES Table
SQL> -- Data dump of TEA provided Sales documentation
SQL> -- TABLESPACE - ADHOC
SQL> -- SOURCE C:\TEA\csv\sales.csv
SQL> -- 13 rows, skip 1
SQL> -- SOURCE NOTES - sales tab of provided Excel file
SQL> --           - changed format and date of some date fields (invalid dates, etc)
SQL> -- After table creation run in c:\
SQL> -- SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\adhocsalescontrol.ctl
log=C:\tea\csv\control\controllogs\adhocsalescontrol.log bad=C:\tea\csv\control\controllogs\badadhocsales.log
SQL> -----
SQL>
SQL>
SQL>
SQL> CREATE TABLE adhoc_allsales
2   (saledate DATE,

```

```

3  cust_id number(3),
4  itinerary_num number(5),
5  agent varchar2 (2),
6  booking_num  varchar2(10),
7  product_cat number(3),
8  supplier_id number(5),
9  Supp_office number(1),
10 trip_start DATE,
11 trip_end DATE,
12 travelclass varchar2 (5),
13 traveller_count number(2),
14 product varchar2 (30),
15 product_des varchar2 (40),
16 destination varchar2 (30),
17 dest_id varchar2 (10),
18 credit_card varchar2 (10),
19 credit_card_exp DATE,
20 credit_card_num number(20),
21 bill_date DATE ,
22 bill_des varchar2 (15),
23 baseprice number(10),
24 totalpricewtax number(10),
25 billedamt number(10),
26 agencyfee varchar2 (15),
27 feeamt number(10),
28 agencycomm number(10))
29 TABLESPACE adhoc
30 STORAGE (initial 50k);

```

Table created.

SQL>

SQL>

SQL> desc adhoc_allclient;

Name	Null?	Type
CUST_ID		NUMBER(3)
FIRSTNAME		VARCHAR2(20)
LASTNAME		VARCHAR2(20)
AGENT		VARCHAR2(2)
EMAIL		VARCHAR2(25)
HOMEPHONE		NUMBER(12)
BUSINESSPHONE		NUMBER(10)
BIRTHDATE		DATE
ADDRESS		VARCHAR2(35)
CITY		VARCHAR2(10)
POSTALCODE		VARCHAR2(7)
PROV		VARCHAR2(2)
COUNTRY		VARCHAR2(10)
COMMENTS		VARCHAR2(50)

```
SQL>
SQL>
SQL>
SQL> SET ECHO OFF
-----
-- SQLLDR COMMANDS (manually pasted in here from CMD)
--
-- Manually formatted to fit on one page for print and readability
--
-- Log files, and Control Files attached, further data count verification on next spool
-----

-- Adhoc_AllClient

C:\Windows\System32>SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\adhocclientcontrol.ctl
log=C:\tea\csv\control\controllogs\adhocclientcontrol.log bad=C:\tea\csv\control\controllogs\badadhocclient.log

SQL*Loader: Release 11.2.0.1.0 - Production on Fri Jan 6 15:50:05 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Load completed - logical record count 270.

-- Adhoc_AllSupplier

C:\Windows\System32>SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\adhocsuppliercontrol.ctl
log=C:\tea\csv\control\controllogs\adhocsuppliercontrol.log bad=C:\tea\csv\control\controllogs\badadhocsupplier.log

SQL*Loader: Release 11.2.0.1.0 - Production on Fri Jan 6 15:52:05 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Load completed - logical record count 1195.

-- Adhoc_AllSupplier

C:\Windows\System32>SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\adhocsalescontrol.ctl
log=C:\tea\csv\control\controllogs\adhocsalescontrol.log bad=C:\tea\csv\control\controllogs\badadhocsales.log
```

```
SQL*Loader: Release 11.2.0.1.0 - Production on Fri Jan 6 15:56:44 2017
Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Load completed - logical record count 1341.
```

```
SQL> SET ECHO ON
SQL> SET VERIFY ON
SQL> TTITLE OFF
SQL>
SQL> -- Summary of cleanup.
SQL> --
SQL> -- The product table primary key is the composite between supplier_ref, region_ref, and product_cat. Some values have
SQL> -- duplicates which are the travel product descriptions. There is a business need to increment the duplicate products
SQL> -- by one, if they occur, by supplier.
SQL> --
SQL> -- We need to import these values in a temporary table, which will allow us to tag a sequence to them for the
SQL> -- only reason - updating the value product category ("Update where sequence value is ..").
SQL> -- Prep table, temp, no constraints
SQL>
SQL> CREATE TABLE teadmin.producttemp2
  2      (update_id number (5),
  3       product_cat number(5),
  4       supplier_ref number(5),
  5       region_ref number(1),
  6       product_des varchar2(30))
  7   TABLESPACE COMPANYDATA
  8       PCTFREE 20
  9       PCTUSED 60
 10      INITTRANS 2
 11      STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);
```

Table created.

```
SQL>
SQL>
SQL>
SQL> -- The (SELECT MIN(ROWID) code below I sourced from stack overflow, which discussed deleting duplicate data by selecting
any data older than
SQL> -- the first occurrence of the data. When classmates were confused about this, I had to share. It's too good a trick.
The rest of the code
SQL> -- mine though!
SQL> -- Source - http://stackoverflow.com/questions/529098/removing-duplicate-rows-from-table-in-oracle
SQL> --
SQL> --
```

```

SQL> -- Check duplicate data, separate unique records for master list
SQL> -- Not In works to find duplicates we may want to delete as opposed to distinct (cannot select 'not distinct')
SQL>
SQL> SELECT supplier_id, supp_office,product_cat
  2 FROM teadmin.adhoc_allsupplier
  3 WHERE rowid not in
  4       (SELECT MIN(rowid)
  5        FROM teadmin.adhoc_allsupplier
  6        GROUP BY supplier_id, supp_office,product_cat)ORDER BY supplier_id;

```

SUPPLIER_ID	SUPP_OFFICE	PRODUCT_CAT
168	1	500
168	1	400
168	1	350
1416	1	251
1416	1	500
1416	1	501
1416	1	500
2099	1	100
2099	1	500
2966	1	452
2966	1	451
3600	1	500
3600	1	400
3631	1	500
5827	1	350
7075	1	150
7075	1	150
7075	1	150
7075	1	150
7075	1	150
7453	2	500

21 rows selected.

```

SQL>
SQL> --Duplicate product_cat with issues
SQL> -- 3631, 7075, 7453, 168, 2099, 3600, 1416, 5827, 2966
SQL>
SQL> --Further investigation, checking each of the ref's above which were pulled 'distinctly'
SQL>
SQL> SELECT count(*)
  2 FROM adhoc_allsupplier
  3 WHERE supplier_id IN (3631, 7075, 7453, 168, 2099, 3600, 1416, 5827, 2966)
  4 ORDER BY supplier_id, supp_office, product_cat;

```

COUNT(*)
70

```

SQL> --70 rows!
SQL> -- As per documentation from TEA, we will increment the product keys for matching supplier, and supplier reference
numbers
SQL>
SQL> -- Get data imported to temp table.
SQL>
SQL> drop sequence teaadmin.updateprod_cat_seq;
drop sequence teaadmin.updateprod_cat_seq
      *
ERROR at line 1:
ORA-02289: sequence does not exist

```

```

SQL>
SQL> create sequence teaadmin.update_prod_cat_seq
  2 increment by 1
  3 nomaxvalue
  4 start with 999
  5 minvalue 1 ;
Sequence created.

```

```

SQL>
SQL>
SQL> INSERT INTO teaadmin.producttemp2 (update_id, product_cat, supplier_ref, region_ref, product_des)
  2 SELECT teaadmin.update_prod_cat_seq.nextval, product_cat, supplier_id, supp_office, prod_des FROM adhoc_allsupplier ;
1193 rows created.

```

```

SQL>
SQL>
SQL> --Select some random data. This code I discovered last exam, from stack overflow. Very cool.
SQL>
SQL> SELECT update_id, product_cat, supplier_ref, region_ref, product_des
  2 FROM (
  3   SELECT *
  4   FROM producttemp2
  5   ORDER BY DBMS_RANDOM.RANDOM)
  6 WHERE rownum < 5;

```

UPDATE ID	PRODUCT CAT	SUPPLIER REF	REGION REF	PRODUCT DES
8630	400	3600	1	MOTOR COACH TOUR OPER
8921	300	6881	1	CRUISE LINES
8760	300	1713	1	CRUISE LINES
8835	500	1900	1	TOUR OPERATORS/WHOLES

```

SQL>
SQL> commit;

Commit complete.

SQL>
SQL> -- Verified
SQL> -- Back to problem data
SQL>
SQL> --count
SQL>
SQL> SELECT count(product_cat)
  2  FROM producttemp2
  3 WHERE supplier_ref IN (3631, 7075, 7453, 168, 2099, 3600, 1416, 5827, 2966)
  4 ORDER BY supplier_ref, region_ref, product_cat;

COUNT (PRODUCT_CAT)
-----
70

SQL>
SQL> -- 70 rows
SQL>
SQL> --Select sample of problem
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat, product_des
  2  FROM producttemp2
  3 WHERE supplier_ref IN (3631);

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT PRODUCT_DES
-----
8164      3631          1        400 MOTOR COACH TOUR OPER
8165      3631          1        500 TOUR OPERATORS/WHOLES
8188      3631          1        500 TOUR OPERATORS/WHOLES

SQL>
SQL> -- Start editing product_cat
SQL> -- Gather master list of changes
SQL>
SQL> set pagesize 100
SQL> BREAK ON supplier_ref
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2  FROM producttemp2
  3 WHERE supplier_ref IN (3631, 7075, 7453, 168, 2099, 3600, 1416, 5827, 2966)
  4 ORDER BY supplier_ref, region_ref, product_cat;

```

UPDATE_ID	SUPPLIER_REF	REGION_REF	PRODUCT_CAT
8236	168	1	350
8412		1	350
8410		1	400
9247		1	400
9248		1	500
8411		1	500
9181	1416	1	100
9218		1	101
8389		1	102
9182		1	250
9333		1	251
8692		1	251
8220		1	252
8523		1	253
8275		1	254
8565		1	255
9183		1	300
9055		1	301
9056		1	302
8773		1	303
8518		1	304
8473		1	305
8987		1	306
9106		1	307
8915		1	308
8462		1	309
9151		1	310
9184		1	350
8390		1	351
8693		1	500
9185		1	500
9110		1	500
8391		1	501
9219		1	501
8280	2099	1	100
8531		1	100
8532		1	500
8281		1	500
9211	2966	1	100
9212		1	300
9347		1	301
8537		1	302
9145		1	303
9143		1	304
9213		1	450
9040		1	451
9060		1	451
9290		1	452
9291		1	452
8949		1	453
9214		1	500

8630	3600	1	400
8460		1	400
8461		1	500
8631		1	500
8164	3631	1	400
8165		1	500
8188		1	500
9027	5827	1	350
9026		1	350
8998		1	351
8879	7075	1	150
9350		1	150
8214		1	150
8191		1	150
8808		1	150
9053		1	150
9109		1	500
8211	7453	2	500
8212		2	500

70 rows selected.

```
SQL> CLEAR BREAKS
SQL>
SQL> -- Spool off, begin manual copy paste to cmd - spooled and appended (just not run as script)
SQL> -- Here we go..
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL> --Supplier 168
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2  FROM producttemp2 WHERE supplier_ref = 168
  3  ORDER BY product_cat;
```

UPDATE_ID	SUPPLIER_REF	REGION_REF	PRODUCT_CAT
8236	168	1	350
8412	168	1	350
9247	168	1	400
8410	168	1	400
9248	168	1	500
8411	168	1	500

6 rows selected.

```
SQL>
SQL>
SQL> UPDATE producttemp2 SET product_cat = 350+1 WHERE update_id = 8412;
```

```

1 row updated.

SQL> UPDATE producttemp2 SET product_cat =400+1 WHERE update_id = 9047;
1 row updated.

SQL> UPDATE producttemp2 SET product_cat =500+1 WHERE update_id = 8411;
1 row updated.

SQL>
SQL> --Supplier 1416
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2   FROM producttemp2 WHERE supplier_ref = 1416
  3   ORDER BY product_cat;

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
----- ----- ----- -----
 9181      1416        1      100
 9218      1416        1      101
 8389      1416        1      102
 9182      1416        1      250
 9333      1416        1      251
 8692      1416        1      251
 8220      1416        1      252
 8523      1416        1      253
 8275      1416        1      254
 8565      1416        1      255
 9183      1416        1      300
 9055      1416        1      301
 9056      1416        1      302
 8773      1416        1      303
 8518      1416        1      304
 8473      1416        1      305
 8987      1416        1      306
 9106      1416        1      307
 8915      1416        1      308
 8462      1416        1      309
 9151      1416        1      310
 9184      1416        1      350
 8390      1416        1      351
 9110      1416        1      500
 9185      1416        1      500
 8693      1416        1      500
 8391      1416        1      501
 9219      1416        1      501

28 rows selected.

SQL>
SQL> UPDATE producttemp2 SET product_cat =250+6 WHERE update_id = 8692;

```

```

1 row updated.

SQL> UPDATE producttemp2 SET product_cat =500+2 WHERE update_id = 9219;
1 row updated.

SQL> UPDATE producttemp2 SET product_cat =500+3 WHERE update_id = 9185;
1 row updated.

SQL> UPDATE producttemp2 SET product_cat =500+4 WHERE update_id = 9110;
1 row updated.

SQL>
SQL> --Supplier 2099
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2  FROM producttemp2 WHERE supplier_ref = 2099
  3  ORDER BY product_cat;

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
----- ----- ----- -----
 8531      2099        1      100
 8280      2099        1      100
 8532      2099        1      500
 8281      2099        1      500

SQL>
SQL> UPDATE producttemp2 SET product_cat =100+1 WHERE update_id = 8531;
1 row updated.

SQL> UPDATE producttemp2 SET product_cat =500+1 WHERE update_id = 8281;
1 row updated.

SQL>
SQL> --2966
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2  FROM producttemp2 WHERE supplier_ref = 2966
  3  ORDER BY product_cat;

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
----- ----- ----- -----
 9211      2966        1      100

```

9212	2966	1	300
9347	2966	1	301
8537	2966	1	302
9145	2966	1	303
9143	2966	1	304
9213	2966	1	450
9060	2966	1	451
9040	2966	1	451
9291	2966	1	452
9290	2966	1	452
8949	2966	1	453
9214	2966	1	500

13 rows selected.

```
SQL>
SQL> UPDATE producttemp2 SET product_cat =450+2 WHERE update_id = 9060;
```

1 row updated.

```
SQL> UPDATE producttemp2 SET product_cat =450+2 WHERE update_id = 9291;
```

1 row updated.

```
SQL>
SQL> --Supplier 3600
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2  FROM producttemp2 WHERE supplier_ref = 3600
  3  ORDER BY product_cat;
```

UPDATE_ID	SUPPLIER_REF	REGION_REF	PRODUCT_CAT
8630	3600	1	400
8460	3600	1	400
8631	3600	1	500
8461	3600	1	500

```
SQL>
SQL> UPDATE producttemp2 SET product_cat =400+1 WHERE update_id = 8460;
```

1 row updated.

```
SQL> UPDATE producttemp2 SET product_cat =500+1 WHERE update_id = 8631;
```

1 row updated.

```

SQL>
SQL> --Supplier 3631
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2   FROM producttemp2 WHERE supplier_ref = 3631
  3 ORDER BY product_cat;

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
----- -----
  8164        3631          1        400
  8188        3631          1        500
  8165        3631          1        500

SQL>
SQL> UPDATE producttemp2 SET product_cat =500+1 WHERE update_id = 8631;

1 row updated.

SQL>
SQL>
SQL> --Supplier 5827
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2   FROM producttemp2 WHERE supplier_ref = 5827
  3 ORDER BY product_cat;

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
----- -----
  9026        5827          1        350
  9027        5827          1        350
  8998        5827          1        351

SQL>
SQL> UPDATE producttemp2 SET product_cat =350+1 WHERE update_id = 9026;

1 row updated.

SQL>
SQL>
SQL> -- Supplier 7075
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2   FROM producttemp2 WHERE supplier_ref = 7075
  3 ORDER BY product_cat;

```

```

UPDATE ID SUPPLIER REF REGION REF PRODUCT CAT
-----
 8191      7075      1      150
 8214      7075      1      150
 8808      7075      1      150
 9350      7075      1      150
 9053      7075      1      150
 8879      7075      1      150
    9109      7075      1      500

7 rows selected.

SQL>
SQL> UPDATE producttemp2 SET product_cat =150+1 WHERE update_id = 9350;
1 row updated.

SQL> UPDATE producttemp2 SET product_cat =150+2 WHERE update_id = 8214;
1 row updated.

SQL> UPDATE producttemp2 SET product_cat =150+3 WHERE update_id = 8191;
1 row updated.

SQL> UPDATE producttemp2 SET product_cat =150+4 WHERE update_id = 8808;
1 row updated.

SQL> UPDATE producttemp2 SET product_cat =150+5 WHERE update_id = 9053;
1 row updated.

SQL>
SQL>
SQL> -- Supplier 7453
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2  FROM producttemp2 WHERE supplier_ref = 7453
  3  ORDER BY product_cat;

UPDATE ID SUPPLIER REF REGION REF PRODUCT CAT
-----
 8212      7453      2      500
 8211      7453      2      500

SQL>
SQL> UPDATE producttemp2 SET product_cat =500+1 WHERE update_id = 8212;

```

```

1 row updated.

SQL>
SQL> -- Any more?
SQL>
SQL> SELECT supplier_ref,region_ref,product_cat
  2  FROM teaadmin.producttemp2
  3 WHERE rowid not in
  4       (SELECT MIN(rowid)
  5        FROM teaadmin.producttemp2
  6        GROUP BY supplier_ref, region_ref ,product_cat)ORDER BY supplier_ref, product_cat;

SUPPLIER_REF REGION_REF PRODUCT_CAT
-----
 168          1      400
 2966         1      452
 2966         1      452
 3631         1      500
 5827         1      351

SQL> commit;

Commit complete.

SQL> --supplier 168, missed an update
SQL> --Supplier 168
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2  FROM producttemp2 WHERE supplier_ref = 168
  3 ORDER BY product_cat;

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
-----
 8236        168      1      350
 8412        168      1      351
 8410        168      1      400
 9247        168      1      400
 9248        168      1      500
 8411        168      1      501

6 rows selected.

SQL> UPDATE producttemp2 SET product_cat =400+1 WHERE update_id = 9247;

1 row updated.

SQL> -- Supplier 2966 missed an update
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2  FROM producttemp2 WHERE supplier_ref = 2966
  3 ORDER BY product_cat;

```

```

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
-----
 9211      2966        1      100
 9212      2966        1      300
 9347      2966        1      301
 8537      2966        1      302
 9145      2966        1      303
 9143      2966        1      304
 9213      2966        1      450
 9040      2966        1      451
 9060      2966        1      452
 9291      2966        1      452
 9290      2966        1      452
 8949      2966        1      453
 9214      2966        1      500

```

13 rows selected.

```
SQL> UPDATE producttemp2 SET product_cat = 450+3 WHERE update_id = 9291;
```

1 row updated.

```
SQL> UPDATE producttemp2 SET product_cat = 450+4 WHERE update_id = 9290;
```

1 row updated.

```
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
 2   FROM producttemp2 WHERE supplier_ref = 3631
 3 ORDER BY product_cat;
```

```

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
-----
 8164      3631        1      400
 8188      3631        1      500
 8165      3631        1      500

```

```
SQL> UPDATE producttemp2 SET product_cat = 500+1 WHERE update_id = 8165;
```

1 row updated.

```
SQL>
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
 2   FROM producttemp2 WHERE supplier_ref = 5827
 3 ORDER BY product_cat;
```

```

UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
-----
 9027      5827        1      350
 8998      5827        1      351

```

```
9026      5827      1      351
```

```
SQL> UPDATE producttemp2 SET product_cat = 350+2 WHERE update_id = 9026;
```

```
1 row updated.
```

```
SQL>
```

```
SQL> -- Any more?
```

```
SQL>
```

```
SQL> SELECT supplier_ref, region_ref, product_cat
  2  FROM teaadmin.producttemp2
  3 WHERE rowid not in
  4       (SELECT MIN(rowid)
  5        FROM teaadmin.producttemp2
  6        GROUP BY supplier_ref, region_ref ,product_cat)ORDER BY supplier_ref, product_cat;
```

```
SUPPLIER REF REGION REF PRODUCT CAT
```

```
----- ----- -----
2966      1      453
```

```
SQL>
```

```
SQL>
```

```
SQL> --2966
```

```
SQL>
```

```
SQL> SELECT update_id, supplier_ref, region_ref, product_cat
  2  FROM producttemp2 WHERE supplier_ref = 2966
  3 ORDER BY product_cat;
```

```
UPDATE_ID SUPPLIER_REF REGION_REF PRODUCT_CAT
```

```
----- ----- -----
9211      2966      1      100
9212      2966      1      300
9347      2966      1      301
8537      2966      1      302
9145      2966      1      303
9143      2966      1      304
9213      2966      1      450
9040      2966      1      451
9060      2966      1      452
8949      2966      1      453
9291      2966      1      453
9290      2966      1      454
9214      2966      1      500
```

```
13 rows selected.
```

```
SQL> UPDATE producttemp2 SET product_cat = 450+5 WHERE update_id = 9291;
```

```
1 row updated.
```

```
SQL> SELECT supplier_ref,region_ref,product_cat
  2  FROM teaadmin.producttemp2
  3 WHERE rowid not in
  4       (SELECT MIN(rowid)
  5        FROM teaadmin.producttemp2
  6        GROUP BY supplier_ref, region_ref ,product_cat)ORDER BY supplier_ref, product_cat;

no rows selected

SQL> -- Clean Data
SQL> spool off

SQL> SET ECHO ON
SQL> SET VERIFY ON
SQL> TTITLE OFF
SQL>
SQL>
SQL> drop table teaadmin.employee cascade constraints;

Table dropped.

SQL> drop table teaadmin.employeedetail cascade constraints;

Table dropped.

SQL> drop table teaadmin.cc cascade constraints;

Table dropped.

SQL> drop table teaadmin.customer cascade constraints;

Table dropped.

SQL> drop table teaadmin.customercomm cascade constraints;

Table dropped.

SQL> drop table teaadmin.customerdetail cascade constraints;

Table dropped.

SQL> drop table teaadmin.salary cascade constraints;

Table dropped.
```

```

SQL> -- EMPLOYEE TABLE
SQL> -- Parent table to employeeDetail and salary
SQL>
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE C:\TEA\csv\employee.csv
SQL> -- 13 rows, skip 1
SQL> -- SOURCE NOTES - data from Tea Docs\tablecodes.doc (provided by TEA)
SQL> --           - employee ID has been generated and added to this document
SQL> --
SQL> -- After table creation run in c:\ 
SQL> -- SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\employeecontrol.ctl
log=C:\tea\csv\control\controllogs\empcontrol.log bad=C:\tea\csv\control\controllogs\bademp.log
SQL> -----
SQL>
SQL>
SQL> CREATE TABLE teaadmin.employee
  2  (emp_ref number(3) CONSTRAINT empref_pk PRIMARY KEY deferrable initially immediate
  3          USING INDEX (CREATE INDEX teaadmin.empref_idx
  4                      ON teaadmin.employee(emp_ref)
  5                      PCTFREE 10 INITTRANS 2
  6                      STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1
  7                      MAXEXTENTS 40 PCTINCREASE 5 )
  8                      TABLESPACE INDX),
  9      first_name varchar2(20),
10      last_name varchar2(20))
11  TABLESPACE COMPANYDATA
12  PCTFREE 20
13  PCTUSED 60
14  INITTRANS 2
15  STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```
SQL> COMMENT ON TABLE teaadmin.employee IS 'TEA Employee Records - Mgmt Access Only';
```

Comment created.

```

SQL>
SQL> -- VERIFY CREATE
SQL>
SQL> DESC teaadmin.employee;
      Name          Null?    Type
-----  -----
EMP_REF           NUMBER(3)
FIRST_NAME        VARCHAR2(20)
LAST_NAME         VARCHAR2(20)

SQL>
SQL>
SQL>
SQL> -----
SQL> -- EMPLOYEE DETAILS TABLE
SQL> -- Child table of employee
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE C:\TEA\csv\employeedetail.csv
SQL> -- 13 rows, skip 1
SQL> -- SOURCE NOTES - Assumed data for employee contact/detail information
SQL> --           - created for mock up purposes
SQL> --
SQL> -- After table creation run in c:\
SQL> -- SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\employeedetailcontrol.ctl
log=C:\tea\csv\control\controllogs\empdetailcontrol.log bad=C:\tea\csv\control\controllogs\badempdetail.log
SQL> -----
SQL>
SQL>
SQL>
SQL> CREATE TABLE teaadmin.employeedetail
  2  (emp_ref number(3) CONSTRAINT emp_pk PRIMARY KEY deferrable initially immediate
  3          USING INDEX (CREATE INDEX teaadmin.emp_idx
  4                                ON teaadmin.employeedetail(emp_ref)
  5                                PCTFREE 10 INITTRANS 2
  6                                STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1
  7                                MAXEXTENTS 40 PCTINCREASE 5 )
  8          TABLESPACE INDX),
  9  prevAgentID varchar2(2),
10  hire_date date,
11  position varchar2(20),
12  address varchar2(40),
13  phone_number number(10),
14  emergency_contact varchar2(10),
15  emergency_contact_phone number(10),
16  comm varchar2(50),
17  CONSTRAINT empdet_emp_fk FOREIGN KEY (emp_ref) REFERENCES employee (emp_ref) deferrable initially immediate)

```

```

18  TABLESPACE COMPANYDATA
19  PCTFREE 20
20  PCTUSED 60
21  INITTRANS 2
22  STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```
SQL> COMMENT ON TABLE teaadmin.employeedetail IS 'TEA Employee Data';
```

Comment created.

```
SQL>
SQL> VERIFY CREATE
SP2-0734: unknown command beginning "VERIFY CRE..." - rest of line ignored.
```

```
SQL>
SQL> DESC teaadmin.employeedetail;
```

Name	Null?	Type
EMP_REF		NUMBER(3)
PREVAGENTID		VARCHAR2(2)
HIRE_DATE		DATE
POSITION		VARCHAR2(20)
ADDRESS		VARCHAR2(40)
PHONE_NUMBER		NUMBER(10)
EMERGENCY_CONTACT		VARCHAR2(10)
EMERGENCY_CONTACT_PHONE		NUMBER(10)
COMM		VARCHAR2(50)

```
SQL>
SQL> -----
SQL> -- SALARY TABLE
SQL> -- Child table of employee
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE C:\TEA\csv\empsal.csv
SQL> -- 13 rows, skip 1
SQL> -- SOURCE NOTES - Assumed data for employee salary information
SQL> --           - created for mock up purposes
SQL> --
SQL> -- After table creation run in c:\
SQL> -- SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\employeesalarycontrol.ctl
log=C:\tea\csv\control\controllogs\empsalarycontrol.log bad=C:\tea\csv\control\controllogs\badempsalary.log
SQL> -----
SQL>
SQL>
SQL> CREATE TABLE teaadmin.salary
2  (emp_ref number(3) CONSTRAINT salary_pk PRIMARY KEY deferrable initially immediate
```

```

3      USING INDEX (CREATE INDEX teaadmin.salary_idx
4                      ON teaadmin.salary(emp_ref)
5                      PCTFREE 10 INITTRANS 2
6                      STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1
7                      MAXEXTENTS 40 PCTINCREASE 5 )
8      TABLESPACE INDX),
9 lastreviewdate date,
10 currmonthsal number(7,2) constraint sal_nn not null,
11 prevmonthsal number(7,2),
12 increasereason varchar2(70),
13 CONSTRAINT empsal_emp_fk FOREIGN KEY (emp_ref) REFERENCES employee (emp_ref) deferrable initially immediate)
14 TABLESPACE COMPANYDATA
15 PCTFREE 20
16 PCTUSED 60
17 INITTRANS 2
18 STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

SQL> COMMENT ON TABLE teaadmin.salary IS 'TEA Employee Salary Records - Mgmt Access Only';

Comment created.

SQL>

SQL>

SQL> --VERIFY CREATE

SQL>

SQL> DESC teaadmin.salary;

Name	Null?	Type
EMP_REF		NUMBER(3)
LASTREVIEWDATE		DATE
CURRMONTHSAL	NOT NULL	NUMBER(7,2)
PREVMONTHSAL		NUMBER(7,2)
INCREASEREASON		VARCHAR2(70)

```

SQL> Prompt ****
*****
SQL> PROMPT Press Load SQLLDR Employee data, press ENTER when ready
Press Load SQLLDR Employee data, press ENTER when ready
SQL> PROMPT (Control Files C:\TEA\csv\control)
(Control Files C:\TEA\csv\control)
SQL> PROMPT (Control Logs C:\TEA\csv\control\controllogs)
(Control Logs C:\TEA\csv\control\controllogs)
SQL> Prompt ****
*****
SQL> PAUSE

SQL>
SQL> -- Create Employee tables complete
SQL>
SQL> -- SQLLDR EmployeeDetail
SQL>
SQL> --C:\Windows\System32>SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\employeedetailcontrol.ctl
--log=C:\tea\csv\control\contlogs\empdetailcontrol.log bad=C:\tea\csv\control\controllogs\badempdetail.log

SQL> --SQL*Loader: Release 11.2.0.1.0 - Production on Sat Jan 7 16:40:41 2017
SQL> --Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
SQL>
SQL> --Load completed - logical record count 12.
SQL>
SQL> -- SQLLDR Employee
SQL> --
SQL> --C:\Windows\System32> SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\employeecontrol.ctl
--log=C:\tea\csv\control\controllogs\empcontrol.log bad=C:\tea\csv\control\controllogs\bademp.log

SQL> --SQL*Loader: Release 11.2.0.1.0 - Production on Sat Jan 7 16:37:46 2017
SQL> --Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
SQL>
SQL> --Load completed - logical record count 12.
SQL>
SQL> -- SQLLDR Salary
SQL>
SQL> -- C:\Windows\System32>SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\employeesalarycontrol.ctl
-- log=C:\tea\csv\control\controllogs\empsalarycontrol.log bad=C:\tea\csv\control\controllogs\badempsalary.log

SQL> -- SQL*Loader: Release 11.2.0.1.0 - Production on Sat Jan 7 16:42:25 2017
SQL> -- Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
SQL>

```

```
SQL> --Load completed - logical record count 12.  
SQL>  
SQL> select count (*) from employee;
```

COUNT(*)

12

```
SQL> select count (*) from employeedetail;
```

COUNT(*)

12

```
SQL> select count (*) from salary;
```

COUNT(*)

12

```
--Control files included in Appendix
```

```

SQL>
SQL>
SQL> --Random Employee Report for data verification
SQL>
SQL> set linesize 100
SQL> ttitle 'Employee Review'
SQL> col emp_ref format 999
SQL> col last_name format a15
SQL> col previousagentid format a6
SQL> col increasereason format a20
SQL>
SQL> SELECT *
  2   FROM (
  3     SELECT e.emp_ref, e.last_name, ed.prevagentid, ed.position, s.lastreviewdate, s.currmonthsal, s.increasereason
  4     FROM teaadmin.employee e JOIN employeedetail ed ON e.emp_ref = ed.emp_ref JOIN salary s ON ed.emp_ref = s.emp_ref
  5    ORDER BY DBMS_RANDOM.RANDOM)
  6   WHERE rownum < 5
  7   order by emp_ref;

```

Sat Jan 07

page 1

Employee Review

EMP_REF	LAST_NAME	PR	POSITION	LASTREVIE	CURRMONTHSAL	INCREASEREASON
104	Dahl	JD	Agent		1350	
107	Merrill	JM	Agent		1400	
108	Peterson	BP	Agent		1000	
110	W			11-OCT-15	1700	Satisfactory Review. Pay Increase.

```

SQL>
SQL> ttitle off
SQL>
SQL>
SQL>
SQL>
SQL> commit;

```

Commit complete.

```

SQL> -----
SQL> -- CUSTOMER TABLE
SQL> -- Parent table of customerdetail, customercomm, credit_card
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.adhoc_allclient
SQL> -- SOURCE NOTES - Verify no duplicate customer_ref numbers in customer adhoc table
SQL>
SQL> SELECT count(*)
  2  FROM adhoc_allclient;

      COUNT(*)
-----
      270

SQL>
SQL> SELECT DISTINCT (count(*))
  2  FROM adhoc_allclient;

      (COUNT(*))
-----
      270

SQL>
SQL> -- No need to worry about filtering duplicates
SQL> -- 270 lines total and 270 distinct
SQL> -----
SQL>
SQL> CREATE TABLE teaadmin.customer
  2  (cust_ref number(3),
  3  first_name varchar(20),
  4  last_name varchar(20),
  5  preferred_agent varchar2(2),
  6  CONSTRAINT cust_ref_pk PRIMARY KEY (cust_ref) deferrable initially immediate USING INDEX
  7  (CREATE INDEX cust_ref_idx ON teaadmin.customer (cust_ref)
  8    PCTFREE 10 INITTRANS 2
  9    STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
 10   TABLESPACE indx))
 11 TABLESPACE COMPANYDATA
 12   PCTFREE 20
 13   PCTUSED 60
 14   INITTRANS 2
 15   STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

Table created.

SQL>

```

```

SQL> INSERT INTO teaadmin.customer
  2  (cust_ref, first_name, last_name, preferred_agent)
  3  SELECT cust_id, firstname, lastname, agent
  4  FROM teaadmin.adhoc_allclient;

270 rows created.

SQL>
SQL> COMMENT ON TABLE teaadmin.customer IS 'Distinct Customer Detail List';

Comment created.

SQL>
SQL>
SQL> -----
SQL> -- CUSTOMERDETAIL TABLE
SQL> -- Child table of Customer
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.adhoc_allclient
SQL> --
SQL> -----
SQL>
SQL> CREATE TABLE teaadmin.customerdetail
  2  (cust_ref number(3) CONSTRAINT cust_det_fk REFERENCES customer(cust_ref) deferrable initially immediate,
  3  home_phone number (12),
  4  business_phone number (12),
  5  address varchar2(35),
  6  city varchar2(10),
  7  prov varchar2(2),
  8  country varchar2(10),
  9  postalcode varchar2(7),
 10 email varchar2 (25),
 11 birth_date DATE,
 12 significant_date DATE,
 13 significant_date_des varchar2(20),
 14 rewards_number number(15) ,
 15 comments varchar2(50),
 16 CONSTRAINT cust_det_ref_pk PRIMARY KEY (cust_ref) deferrable initially immediate USING INDEX
 17          (CREATE INDEX cust_ref_det_idx ON teaadmin.customerdetail (cust_
 18                      _ref)
 19                      PCTFREE 10 INITTRANS 2
 20                      STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
 21                      TABLESPACE indx))
 22 TABLESPACE COMPANYDATA
 23          PCTFREE 20
 24          PCTUSED 60
 25          INITTRANS 2
 26          STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```
SQL> COMMENT ON TABLE teaadmin.customerdetail IS 'Distinct Customer Details List';
```

Comment created.

```
SQL>
```

```
SQL> INSERT INTO teaadmin.customerdetail
  2  (cust_ref, home_phone, business_phone, address, city, prov, country, postalcode, email, birth_date, comments)
  3  SELECT cust_id, homephone, businessphone, address, city, prov, country, postalcode, email, birthdate, comments
  4  FROM teaadmin.adhoc_allclient;
```

270 rows created.

```
SQL>
```

```
SQL> -- Sure wish 11g allowed Create Table As Selects with Referencial Constraints.... the CAST option was really neat!
SQL> -- Using simple insert into instead of Create Table as Select
```

```
SQL>
```

```
SQL> -- VERIFY DATA LOAD
```

```
SQL>
```

```
SQL> select count (*) from customerdetail;
```

COUNT(*)

270

```
SQL>
```

```
SQL> -- 270, expected result
```

```
SQL>
```

```
SQL>
```

```
SQL> -----
```

```
SQL> -- CREDITCARD TABLE
```

```
SQL> -- Child table of Customer
```

```
SQL> --
```

```
SQL> -- TABLESPACE - COMPANYDATA
```

```
SQL> -- SOURCE: teaadmin.adhoc_allsales
```

```
SQL> -- SOURCE NOTES - Source data for creditcards in sales spreadsheet
```

```
SQL> --
```

```
SQL> --
```

```
SQL> -- TABLE NOTES - Keep access locked down on this table
```

```
SQL> -- - Seperate distinct out for master list
```

```
SQL> -----
```

```
SQL>
```

```
SQL>
```

```
SQL> -- Distint creditcards in adhoc_allsales
```

```
SQL>
```

```
SQL> SELECT count(distinct credit_card_num)
```

```

2 from teaadmin.adhoc_allsales;

COUNT(DISTINCTCREDIT_CARD_NUM)
-----
213

SQL>
SQL> -- 213 distinct credit card numbers
SQL>
SQL> SELECT count(distinct credit_card_num||cust_id)
2 FROM adhoc_allsales
3 WHERE rowid >
4 (SELECT MIN(rowid)
5 FROM adhoc_allsales);

COUNT(DISTINCTCREDIT_CARD_NUM||CUST_ID)
-----
215

SQL>
SQL> -- 215 distinct credit cards, and credit card owners..
SQL> -- Credit card being used twice?
SQL>
SQL> -- Will want a second temp credit card table to do data cleanup in
SQL> -- Create table to clean credit card data, want to keep adhoc intact.
SQL>
SQL> CREATE TABLE teaadmin.cctemp
2 ( cust_ref number(3),
3   creditcard number(16),
4   cc_type varchar2(10),
5   exp_date date)
6 TABLESPACE adhoc
7 PCTFREE 20
8 PCTUSED 60
9 INITTRANS 2
10 STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

Table created.

SQL> COMMENT ON TABLE teaadmin.cctemp IS 'TEA Credit Card Data - Limited Access';

Comment created.

SQL>
SQL> INSERT INTO teaadmin.cctemp
2 (cust_ref, creditcard, cc_type, exp_date)
3 SELECT cust_id, credit_card_num, credit_card, credit_card_exp
4 FROM teaadmin.adhoc_allsales;

```

```

1341 rows created.

SQL> -- 1341 rows loaded
SQL> -- All credit card data (including duplicates) loaded to cctemp
SQL>
SQL> -- Delete duplicates, keep credit card number with lowest rowid (Same query used in product category cleanup)
SQL> -- Ensures we keep only unique credit card numbers
SQL>
SQL> DELETE FROM teaadmin.cctemp
  2 WHERE rowid not in
  3 (SELECT MIN(rowid)
  4 FROM teaadmin.cctemp
  5 GROUP BY creditcard, cust_ref);

1126 rows deleted.

SQL>
SQL> --VERIFY NO DUPLICATES
SQL> SELECT creditcard, count (creditcard)
  2 from teaadmin.cctemp
  3 group by creditcard
  4 having count (creditcard) > 1;

  CREDITCARD COUNT(CREDITCARD)
-----
21133458897034          2
78789007977999          2

SQL>
SQL> -- Duplicates showing with 21133458897034 and 78789007977999
SQL> -- With different customer references
SQL>
SQL> select * from teaadmin.cctemp where creditcard in (21133458897034, 78789007977999);

  CUST_REF      CREDITCARD CC_TYPE     EXP_DATE
-----
    105    78789007977999 VISA        22-AUG-14
    104    78789007977999 AMEX        22-AUG-14
    371    21133458897034 Diners     15-AUG-16
    197    21133458897034 Diners     15-AUG-14

SQL>
SQL> --PROBLEM WITH DATA
SQL> --Cust_Ref is different for each card. Possibility the card belongs to two members of same family
SQL> --or data is not correct.
SQL> --I'll make the assumption that creditcard data is not mandatory and will delete offending
SQL> --records.

```

```
SQL> --Will be expecting 211 rows after delete. Can flag customer 371, 197, 105, 104 to update cc info.
SQL>
SQL> delete from teaadmin.cctemp where creditcard in (21133458897034, 78789007977999);
4 rows deleted.

SQL>
SQL> commit;

Commit complete.

SQL>
SQL> --Don't want to overwrite comments, are there any currently for these customers?
SQL>
SQL> select comments from teaadmin.customerdetail where cust_ref in (371,197,105,104);

COMMENTS
-----
 

SQL>
SQL> UPDATE teaadmin.CUSTOMERDETAIL set comments = 'Update Creditcard Info on next visit'
  2 WHERE cust_ref IN (371,197,105,104);

4 rows updated.

SQL>
SQL> select comments from teaadmin.customerdetail where cust_ref in (371,197,105,104);

COMMENTS
-----
Update Creditcard Info on next visit

SQL>
SQL> commit;

Commit complete.
```

```
SQL>
SQL> -- LOAD GOOD CREDIT CARD DATA TO TEAADMIN.CC
SQL>
SQL> CREATE TABLE teaadmin.cc
  2  ( cust_ref number(3) CONSTRAINT cc_cust_fk REFERENCES customer(cust_ref) deferrable initially immediate,
  3    creditcard number(16) CONSTRAINT cc_pk PRIMARY KEY deferrable initially immediate
  4      USING INDEX (CREATE INDEX teaadmin.cc_idx
  5                    ON teaadmin.cc(creditcard)
  6                    PCTFREE 10 INITTRANS 2
  7                    STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1
  8                      MAXEXTENTS 40 PCTINCREASE 5)
  9                    TABLESPACE INDX),
10   cc_type varchar2(10),
11   exp_date date)
12 TABLESPACE COMPANYDATA
13 PCTFREE 20
14 PCTUSED 60
15 INITTRANS 2
16 STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);
```

Table created.

```
SQL>
SQL> INSERT INTO teaadmin.cc (cust_ref, creditcard, cc_type, exp_date)
  2  SELECT cust_ref, creditcard, cc_type, exp_date
  3  FROM teaadmin.cctemp;
```

211 rows created.

```
SQL>
SQL> -- verify data copy
SQL>
SQL> select count(*) from teaadmin.cc;
```

COUNT(*)

211

```
SQL>
SQL> --211 rows, expected result
SQL>
SQL>
SQL> --drop unneeded adhoc table cctemp
SQL>
SQL> drop table cctemp cascade constraints;
```

Table dropped.

```

SQL> purge recyclebin;

Recyclebin purged.

SQL>
SQL>
SQL> -----
SQL> -- CUSTOMERCOMM TEMP TABLE
SQL> -- Child table of Customer
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE:
SQL> -- SOURCE NOTES - Assumed, mock up data
SQL> --           - C:\TEA\csv\customercomm.csv
SQL> --
SQL> -- TABLE NOTES - Loading some 'assumed data' for mock up marketing reports.
SQL> --
SQL> -- Note on Communications Temp Table - Oracle 11g does not support inserting default
SQL> -- sequence.nextval in create table. Will need to create
SQL> -- a temp table to load data with a sequence.
SQL> -- After table creation run at c:\ 
SQL> -- SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\customercomm.ctl
log=C:\tea\csv\control\controllogs\customercomm.log bad=C:\tea\csv\control\controllogs\badcustcomm.log
SQL> -----
SQL>
SQL> -- No constraints on temporary table
SQL>
SQL> CREATE TABLE teaadmin.customercommtemp
  2  (communication_ref number(10),
  3   cust_ref number(3),
  4   lastcontactdate date,
  5   lastcontacttype varchar2(20),
  6   emp_ref number(3))
  7   TABLESPACE ADHOC
  8   PCTFREE 20
  9   PCTUSED 60
10   INITTRANS 2
11   STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```

SQL> create sequence teaadmin.comm_ref_seq
  2 increment by 1
  3 nomaxvalue
  4 start with 999
  5 minvalue 1 ;

```

Sequence Created.

```

SQL>
SQL>
SQL> Prompt ****
*****
SQL> PROMPT Press Load SQLLDR customer communications data, press ENTER when ready
Press Load SQLLDR customer communications data, press ENTER when ready
SQL> PROMPT (Control Files C:\TEA\csv\control)
(Control Files C:\TEA\csv\control)
SQL> PROMPT (Control Logs C:\TEA\csv\control\controllogs)
(Control Logs C:\TEA\csv\control\controllogs)
SQL> Prompt ****
*****
SQL> PAUSE

SQL>
SQL> -- SQLLDR CustomerCommTemp
SQL> --
SQL> --C:\Windows\System32>SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\customercomm.ctl
log=C:\tea\csv\control\controllogs\customercomm.log bad=C:\tea\csv\control\controllogs\badcustcomm.log
SQL> --SQL*Loader: Release 11.2.0.1.0 - Production on Sat Jan 7 17:35:13 2017
SQL> --Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
SQL> --
SQL> --Load completed - logical record count 26.
SQL>
SQL>
SQL>
SQL> -- Add sequence generation to customer communication records
SQL>
SQL> update customercommtemp set communication_ref = comm_ref_seq.nextval;

26 rows updated.

SQL>
SQL> --verify sequence and data additions
SQL>
SQL>
SQL> --Copy from adhoc to proper customer communications table
SQL>
SQL> CREATE TABLE teaadmin.customercomm
  2 (communication_ref number(10) CONSTRAINT custcommref_pk PRIMARY KEY deferrable initially immediate
  3   USING INDEX (CREATE INDEX teaadmin.communication_ref_idx
  4     ON teaadmin.customercomm(communication_ref)
  5       PCTFREE 10 INITTRANS 2
  6       STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1
  7         MAXEXTENTS 40 PCTINCREASE 5 )
  8       TABLESPACE INDX),
  9   cust_ref number(3),

```

```

10 lastcontactdate date,
11 lastcontacttype varchar2(20),
12 emp_ref number(3),
13 CONSTRAINT comm_cust_ref_fk FOREIGN KEY (cust_ref) REFERENCES customer (cust_ref) deferrable initially immediate)
14 TABLESPACE COMPANYDATA
15 PCTFREE 20
16 PCTUSED 60
17 INITTRANS 2
18 STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```
SQL> COMMENT ON TABLE teaadmin.customercomm IS 'TEA Client Contact History - sample data for future tracking';
```

Comment created.

```

SQL>
SQL> -- Insert clean data to it's own proper COMPANYDATA tablespace table
SQL>
SQL> INSERT INTO teaadmin.customercomm
  2  (communication_ref, cust_ref, lastcontactdate, lastcontacttype, emp_ref)
  3  SELECT communication_ref, cust_ref, lastcontactdate, lastcontacttype, emp_ref
  4  FROM teaadmin.customercommtemp;

```

26 rows created.

```

SQL>
SQL> COL agent format 9999
SQL> COL client_name format A15
SQL> TTITLE 'Customer Last Contacted Report'
SQL> SELECT *
  2   FROM (
  3     SELECT cm.communication_ref, cm.cust_ref, c.last_name as ClientName, cm.lastcontactdate, cm.lastcontacttype,
cm.emp_ref as "Agent"
  4     FROM customercomm cm JOIN teaadmin.customer c ON c.cust_ref = cm.cust_ref
  5     ORDER BY DBMS_RANDOM.RANDOM)
  6   WHERE rownum < 5
  7   order by cust_ref;

```

Sat Jan 07

Customer Last Contacted Report

page 1

COMMUNICATION REF	CUST REF	CLIENTNAME	LASTCONTA	LASTCONTACTTYPE	Agent
1031	108	Sethi	15-OCT-15	Cold Call	104
1035	112	Thomas	01-NOV-15	Email	107
1037	114	Laporte	01-NOV-15	Email	107
1040	119	Abdou	15-OCT-15	Cold Call	104

```
SQL>
SQL>
SQL>
SQL> Commit;

Commit complete.

SQL>
SQL> -- Drop customercommtemp table, no need now that data is loaded with sequence.
SQL>
SQL> drop table customercommtemp cascade constraints;

Table dropped.

SQL>
SQL> purge recyclebin;

Recyclebin purged.
```

```
SQL>
SQL>
SQL> -----
SQL> -- FEE TABLE
SQL> -- Child table of Payment
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.adhoc_allsales
SQL> --
SQL> -- TABLE NOTES - Reference list of unique fee codes
SQL> --
SQL> -----
SQL>
SQL>
SQL> select distinct agencyfee from teaadmin.adhoc_allsales;

AGENCYFEE
-----
RS
GR
BK
NC

SQL> -- No conflicting Agency Fee codes in Adhoc_Allsales in current data. Will be making this unique.
```

```
SQL> CREATE TABLE teaadmin.fee
  2   (fee_ref varchar(5) CONSTRAINT fee_ref_pk PRIMARY KEY deferrable initially deferred
  3     USING INDEX (CREATE INDEX fee_ref_idx ON teaadmin.fee (fee_ref)
  4       PCTFREE 10 INITTRANS 2
  5       STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5 )
  6       TABLESPACE indx),
  7   fee_des varchar2(30),
  8   amount number(5,2))
  9 TABLESPACE COMPANYDATA
10   PCTFREE 20
11   PCTUSED 60
12   INITTRANS 2
13   STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);
```

Table created.

```
SQL>
SQL> COMMENT ON TABLE teaadmin.fee IS 'TEA Fee Reference Table';
```

Comment created.

```
SQL>
SQL> select distinct agencyfee from adhoc_allsales;
```

```
AGENCYFEE
-----
RS
GR
BK
NC
```

```
SQL>
SQL>
SQL> -- Taken from TEA tablecodes documentation. Not worth the SQLLDR time...
SQL>
SQL> INSERT INTO teaadmin.fee (fee_ref, fee_des, amount) VALUES ('RF', 'Refund', 25);
```

1 row created.

```
SQL> INSERT INTO teaadmin.fee (fee_ref, fee_des, amount) VALUES ('BK', 'Booking Charge', 25);
1 row created.
```

```
SQL> INSERT INTO teaadmin.fee (fee_ref, fee_des, amount) VALUES ('CH', 'Change', 15);
1 row created.
```

```
SQL> INSERT INTO teaadmin.fee (fee_ref, fee_des, amount) VALUES ('RS', 'Research', 50);
```

```

1 row created.

SQL> INSERT INTO teaadmin.fee (fee_ref, fee_des, amount) VALUES ('GR', 'Group Booking', 100);
1 row created.

SQL> INSERT INTO teaadmin.fee (fee_ref, fee_des, amount) VALUES ('NSF', 'Insufficient Funds', 25);
1 row created.

SQL> INSERT INTO teaadmin.fee (fee_ref, fee_des) VALUES ('NC', 'No Charge' );
1 row created.

SQL>
SQL> -- Verify Data INSERT
SQL>
SQL> SELECT * FROM teaadmin.fee;

FEE_R FEE_DES                AMOUNT
----- -----
RF    Refund                  25
BK    Booking Charge          25
CH    Change                  15
RS    Research                50
GR    Group Booking           100
NSF   Insufficient Funds     25
NC    No Charge               0

7 rows selected.

SQL>
SQL>
SQL> -----
SQL> -- DESTINATION ID TABLE
SQL> -- Child table of Itinerary
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE:
SQL> -- SOURCE NOTES - data from Tea Docs\tablecodes.doc (provided by TEA)
SQL> --             - more cleanup on this file, data in adhoc does not match source data in doc file
SQL> --             - Foreign key in the itinerary parent table.
SQL> --
SQL> --             - 'update' statements - quick job
SQL> --
SQL> -- MAKE DATA IN ADHOC MATCH CODE TABLES
SQL> -- Verify before data
SQL>         select distinct dest_id from teaadmin.adhoc_allsales;

```

```

DEST_ID
-----
MED
SP
AFR
ANZ
MEAST
NA
EU
ASIA
SA

9 rows selected.

SQL>
SQL> --          SP is proper code for South Pacific in tablecodes.doc
SQL>           UPDATE teaadmin.adhoc_allsales
2           SET dest_id = 'SP' where dest_id = 'SPCF';

2 rows updated.

SQL> --
SQL> -----
SQL>
SQL>
SQL> CREATE TABLE teaadmin.destination
2   (dest_id varchar2(5) CONSTRAINT dest_pk PRIMARY KEY deferrable initially immediate
3    USING INDEX (CREATE INDEX teaadmin.dest_id_idx
4                  ON teaadmin.destination(dest_id)
5                  PCTFREE 10 INITTRANS 2
6                  STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1
7                  MAXEXTENTS 40 PCTINCREASE 5 )
8                  TABLESPACE INDX),
9   dest_des varchar2(40)
10  TABLESPACE COMPANYDATA
11  PCTFREE 20
12  PCTUSED 60
13  INITTRANS 2
14  STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

Table created.

SQL> COMMENT ON TABLE teaadmin.destination IS 'Reference Reference Table';

Comment created.

SQL>
SQL>

```

```
SQL> INSERT INTO teaadmin.destination (dest_id)
  2  SELECT DISTINCT dest_id
  3  FROM teaadmin.adhoc_allsales;

9 rows created.

SQL>
SQL> update teaadmin.destination set dest_des = 'Mediterranean' WHERE dest_id = 'MED';

1 row updated.

SQL> update teaadmin.destination set dest_des = 'Australia ''&'' New Zealand' WHERE dest_id = 'ANZ';

1 row updated.

SQL> update teaadmin.destination set dest_des = 'Africa' WHERE dest_id = 'AFR';

1 row updated.

SQL> update teaadmin.destination set dest_des = 'Asia' WHERE dest_id = 'ASIA';

1 row updated.

SQL> update teaadmin.destination set dest_des = 'South America' WHERE dest_id = 'SA';

1 row updated.

SQL> update teaadmin.destination set dest_des = 'Europe'' &'' United Kingdom' WHERE dest_id = 'EU';

1 row updated.

SQL> update teaadmin.destination set dest_des = 'Middle East' WHERE dest_id = 'MEAST';

1 row updated.

SQL> update teaadmin.destination set dest_des = 'South Pacific' WHERE dest_id = 'SP';

1 row updated.

SQL> update teaadmin.destination set dest_des = 'North Pacific' WHERE dest_id = 'NA';

1 row updated.

SQL> update teaadmin.destination set dest_des = 'Other' WHERE dest_id = 'OTHR';

0 rows updated.

SQL>
```

```

SQL> -- No 'other' in current data, adding to master reference list
SQL> INSERT INTO teaadmin.destination (dest_id, dest_des) values ('OTHR', 'Other Destination');

1 row created.

SQL>
SQL> SELECT * FROM teaadmin.destination;

DEST_ DEST_DES
-----
MED Mediterranean
SP South Pacific
AFR Africa
ANZ Australia '&' New Zealand
MEAST Middle East
NA North Pacific
EU Europe' & United Kingdom
ASIA Asia
SA South America
OTHR Other Destination

10 rows selected.

```

```

SQL>
SQL> -----
SQL> -- TRAVEL CLASS TABLE
SQL> -- Child table of Itinerary
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE:
SQL> -- SOURCE NOTES - data from Tea Docs\tablecodes.doc (provided by TEA)
SQL> --           - data provided in document does not all match a SELECT distinct travelclass FROM adhoc_allsales;
SQL> --           - will do a quick clean up in CSV
SQL> --
SQL>           - 388 listings in adhoc_allsales with NULL flightclass/travel class
SQL>           - will load data but not make it a required foreign key reference
SQL> --
SQL>           - Foreign key in the itinerary parent table.
SQL> --
SQL> -- After table creation run in c:\
SQL> -- SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\classtypecontrol.ctl
log=C:\tea\csv\control\controllogs\classtype_control.log bad=C:\tea\csv\control\controllogs\badclasstype.log
SQL> -----
SQL>
SQL>

```

```
SQL> CREATE TABLE teaadmin.travelclass
  2  (travel_class varchar2(5) CONSTRAINT travel_class_pk PRIMARY KEY deferrable initially immediate
  3          USING INDEX (CREATE INDEX teaadmin.travel_class_idx
  4                      ON teaadmin.travelclass(travel_class)
  5                      PCTFREE 10 INITTRANS 2
  6                      STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1
  7                      MAXEXTENTS 40 PCTINCREASE 5 )
  8          TABLESPACE INDX),
  9  class_des varchar2(20))
10 TABLESPACE COMPANYDATA
11 PCTFREE 20
12 PCTUSED 60
13 INITTRANS 2
14 STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);
```

Table created.

```
SQL> COMMENT ON TABLE teaadmin.travelclass IS 'Travel Class Reference Table';
```

Comment created.

```
SQL>
SQL> Prompt ****
*****
SQL> PROMPT Press Load SQLLDR Affiliation data, press ENTER when ready
Press Load SQLLDR Affiliation data, press ENTER when ready
SQL> PROMPT (Control Files C:\TEA\csv\control)
(Control Files C:\TEA\csv\control)
SQL> PROMPT (Control Logs C:\TEA\csv\control\controllogs)
(Control Logs C:\TEA\csv\control\controllogs)
SQL> Prompt ****
*****
SQL> PAUSE

SQL>
SQL> --SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\classtypecontrol.ctl
log=C:\tea\csv\control\controllogs\classtype_control.log bad=C:\tea\csv\control\controllogs\badclasstype.log
```

```
SQL>
```

```
SQL> SELECT * FROM teaadmin.travelclass;
```

```
TRAVE CLASS_DES
-----
FST First Class
BSN Business
ECN Economy
OCNV Ocean View
INT Interior
DLX Deluxe
DBL Double
SNG Single
```

```
8 rows selected.
```

```
SQL> -----
SQL> -- AFFILIATION TABLE
SQL> -- Child table of SupplierDetail
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: Tea Docs\tablecodes.doc
SQL> -- SOURCE NOTES - CSV file created from tablecodes.doc provided by TEA/
SQL> --           - C:\TEA\csv\affiliation.csv
SQL> --
SQL> -- After table creation run at c:\
SQL> -- SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\affiliation.ctl
log=C:\tea\csv\control\controllogs\affiliation.log bad=C:\tea\csv\control\controllogs\badaffiliation.log
SQL> -----
SQL>
SQL> CREATE TABLE teaadmin.Affiliation
  2 (affiliation_code varchar2(10) CONSTRAINT affiliation_code_pk PRIMARY KEY deferrable initially deferred
  3           USING INDEX (CREATE INDEX affiliation_idx
  4                         ON teaadmin.affiliation (affiliation_code)
  5                         PCTFREE 10 INITTRANS 2
  6                         STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5 )
  7           TABLESPACE indx),
  8   affiliation_des varchar2(50))
  9   TABLESPACE COMPANYDATA
10  PCTFREE 20
11  PCTUSED 60
12  INITTRANS 2
13  STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);
```

```
Table created.
```

```
SQL> COMMENT ON TABLE teaadmin.affiliation IS 'TEA Affiliation Data';
```

```
Comment created.
```

```

SQL>
SQL> DESC teaadmin.affiliation;
      Name          Null?    Type
-----  -----
AFFILIATION_CODE                    VARCHAR2(10)
AFFILIATION_DES                     VARCHAR2(50)

SQL>
SQL>
SQL> Prompt ****
***** Press Load SQLLDR Affiliation data, press ENTER when ready
Press Load SQLLDR Affiliation data, press ENTER when ready
SQL> PROMPT (Control Files C:\TEA\csv\control)
(Control Files C:\TEA\csv\control)
SQL> PROMPT (Control Logs C:\TEA\csv\control\controllogs)
(Control Logs C:\TEA\csv\control\controllogs)
SQL> Prompt ****
*****
SQL> PAUSE

SQL>
SQL>
SQL> --Affiliation Data
SQL>
SQL> --C:\>SQLLDR teaadmin/teaadmin control=C:\tea\csv\control\affiliation.ctl
log=C:\tea\csv\control\controllogs\affiliation.log bad=C:\tea\csv\control\controllogs\badaffiliation.log
SQL> --SQL*Loader: Release 11.2.0.1.0 - Production on Sat Jan 7 21:04:40 2017
SQL> --Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
SQL>
SQL> --Load completed - logical record count 7.
SQL>
SQL>
SQL>

```

```

SQL>
SQL> -----
SQL> -- SUPPLIER TABLE
SQL> -- Child table of Supplier
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.adhoc_allsupplier
SQL> --
SQL> -- SOURCE NOTES - Want to only select distinct Supplier list
SQL> --           - Composite primary key on supplier_ref and region_ref
SQL> --           - Excel shows 731 unique supplier_ref, region_ref
SQL> --           - Will also verify in code below
SQL> --
SQL> -----
SQL>
SQL> --Create Supplier parent table
SQL>
SQL> CREATE TABLE teaadmin.supplier
  2  (supplier_ref number (5),
  3  region_ref number (1),
  4  representative_code varchar2(30),
  5  company_name varchar2(75),
  6  comments varchar2(150),
  7  CONSTRAINT supplier_region_pk PRIMARY KEY (supplier_ref, region_ref) deferrable initially immediate USING INDEX
  8          (CREATE INDEX supplier_idx ON teaadmin.supplier (supplier_ref, region_ref) PCTFREE 10 INITTRANS 2
  9            STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
10            TABLESPACE indx))
11          TABLESPACE COMPANYDATA
12 PCTFREE 20
13 PCTUSED 60
14 INITTRANS 2
15 STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```

SQL>
SQL>
SQL> COMMENT ON TABLE teaadmin.supplier IS 'TEA Supplier Data';

```

Comment created.

```

SQL>
SQL> -- Insert all 731 Supplier rows
SQL>
SQL> INSERT INTO teaadmin.supplier
  2  (company_name, supplier_ref, region_ref, representative_code, comments)
  3  SELECT company, supplier_id, supp_office, representative, comments
  4  FROM adhoc_allsupplier

```

```

5      WHERE rowid in
6          (SELECT MIN(rowid)
7           FROM adhoc_allsupplier
8           GROUP BY supplier_id, supp_office);

731 rows created.

SQL>
SQL> SELECT COUNT(*) FROM supplier;

        COUNT(*)
-----
731

SQL>
SQL> commit;

Commit complete.

SQL>
SQL> -----
SQL> -- SUPPLIERDETAIL TABLE
SQL> -- Child table of Supplier
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.adhoc_allsupplier
SQL> --
SQL> -- SOURCE NOTES - Want to only select distinct Supplier information for supplier detail list
SQL> --             - Composite primary key on supplier_ref and region_ref
SQL> --                 - Excel shows 731 unique supplier_ref, region_ref
SQL> --             - Will also verify in code below
SQL> --
SQL> -----
SQL>
SQL> SELECT count (*) FROM adhoc_allsupplier
2 WHERE rowid in
3 (SELECT MIN(rowid)
4 FROM adhoc_allsupplier
5 GROUP BY supplier_id, supp_office);

        COUNT(*)
-----
731

SQL>
SQL> -- Result 731, will create new table for distinct data

```

```

SQL>
SQL> CREATE TABLE teaadmin.supplierdetail
  2      (supplier_ref number(5),
  3       region_ref number(1),
  4       phone_number number(12),
  5       fax_number number(12),
  6       address varchar2(35),
  7       address2 varchar2(30),
  8       city varchar2(25),
  9       prov varchar2(2),
 10      postal_code varchar2(7),
 11      country varchar2(20),
 12      contact_name varchar2(25),
 13      email varchar2(45),
 14      web varchar2(40),
 15      affiliation_code varchar2(15) CONSTRAINT affil_sup_detail_fk REFERENCES affiliation(affiliation_code) deferrable
initially immediate,
 16      comments varchar2(100),
 17      CONSTRAINT details_supp_region_pk PRIMARY KEY (supplier_ref, region_ref) deferrable initially immediate USING INDEX
 18          (CREATE INDEX supplierdetail_idx ON teaadmin.supplierdetail (supplier_ref, region_ref)
 19              PCTFREE 10 INITTRANS 2
 20              STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
 21              TABLESPACE indx),
 22      CONSTRAINT suppdetail_fk FOREIGN KEY (supplier_ref, region_ref) REFERENCES supplier(supplier_ref, region_ref)
deferrable initially immediate)
 23      TABLESPACE COMPANYDATA
 24          PCTFREE 20
 25          PCTUSED 60
 26          INITTRANS 2
 27          STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```

SQL>
SQL> COMMENT ON TABLE teaadmin.supplierdetail IS 'Distinct Supplier details list';

```

Comment created.

```

SQL> INSERT INTO teaadmin.supplierdetail
  2      (supplier_ref, region_ref, phone_number, fax_number, address, address2, city, prov, postal_code, country, contact_name,
email, web, affiliation_code)
  3      SELECT supplier_id, supp_office, phone, fax, supp_address, supp_address2,
  4      city, prov, postalcode, country, contact, email, web,
  5      affiliation
  6      FROM adhoc_allsupplier
  7      WHERE rowid in (SELECT MIN(rowid)    FROM adhoc_allsupplier        GROUP BY supplier_id, supp_office);

```

731 rows created.

```

SQL>
SQL>
SQL> -- VERIFY DATA
SQL> SELECT COUNT (*) FROM supplierdetail;

          COUNT(*)
-----
         731

SQL>
SQL> -- 731 expected result
SQL>
SQL>
SQL> -- Some sample data for futher verification of relationship
SQL> -- random data pull -- join affiliation to details report where description is not null
SQL> -- "Association of Canadian Travel Agents Report"
SQL>
SQL> SELECT supplier_ref, region_ref, affiliation_code, affiliation_des
  2  FROM (
  3    SELECT sd.supplier_ref, sd.region_ref, sd.affiliation_code, a.affiliation_des
  4      FROM teaadmin.supplierdetail sd, teaadmin.affiliation a
  5     WHERE a.affiliation_code = sd.affiliation_code
  6     AND a.affiliation_des IS NOT NULL
  7    ORDER BY DBMS_RANDOM.RANDOM)
  8 WHERE rownum < 5;

SUPPLIER_REF      REGION_REF AFFILIATION_COD AFFILIATION_DES
-----  -----
      2623            1 ACTA        Association of Canadian Travel Agents
      603             1 ACTA        Association of Canadian Travel Agents
     1412            1 ACTA        Association of Canadian Travel Agents
     4994            1 ACTA        Association of Canadian Travel Agents

SQL>
SQL> -- Sample data of supplier data load,
SQL> -- join on supplier details to ensure relationships
SQL>
SQL> COL company_name format A20
SQL> COL contact_name format A20
SQL> SELECT company_name , supplier_ref, region_ref, phone_number,contact_name
  2  FROM (
  3    SELECT s.company_name, s.supplier_ref, s.region_ref, sd.phone_number, sd.contact_name
  4      FROM teaadmin.supplier s, teaadmin.supplierdetail sd
  5     WHERE s.supplier_ref = sd.supplier_ref
  6       AND s.region_ref = sd.region_ref
  7    ORDER BY DBMS_RANDOM.RANDOM)
  8 WHERE rownum < 5;

```

COMPANY NAME	SUPPLIER REF	REGION REF	PHONE NUMBER	CONTACT NAME
SABENA/SWISSAIR SHERATON SALES CENTR E TORONTO	2861 2691	1 1	5149545600 4163611000	Sylvie Leduc
ISSA RESORT COLLECTI ON	5089	1	4166930199	Judy Duncan
KEMWEL HOLIDAY AUTOS	12907	1	9148253100	Gary Hyde

SQL> commit;

Commit complete.

SQL>
SQL> Prompt *****

SQL> PROMPT Supplier create/load complete
Supplier create/load complete
SQL> PROMPT

SQL> PROMPT Press enter to create/load Product table
Press enter to create/load Product table
SQL> Prompt *****

SQL> PAUSE

SQL>
SQL>

```

SQL> -----
SQL> -- PRODUCT TABLE
SQL> -- Parenttable of Supplier
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.producttemp2
SQL> --
SQL> -- SOURCE NOTES - Source is ProductTemp2, which has unique data from adhoc_allsupplier - Clean data!
SQL> --
SQL> -----
SQL>
SQL>
SQL> CREATE TABLE teaadmin.product
  2   (supplier_ref number(5),
  3    region_ref  number(5),
  4    product_cat number(5),
  5    product_des varchar(30),
  6    CONSTRAINT prod_supp_reg_pk PRIMARY KEY (supplier_ref, region_ref, product_cat) USING INDEX
  7      (CREATE INDEX prod_idx ON teaadmin.product (supplier_ref, region_ref, product_cat)
  8       PCTFREE 10 INITTRANS 2
  9       STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
 10      TABLESPACE indx),
 11    CONSTRAINT supp_fk FOREIGN KEY (supplier_ref, region_ref) REFERENCES supplier(supplier_ref, region_ref) deferrable
initially immediate)
 12   TABLESPACE COMPANYDATA
 13     PCTFREE 20
 14     PCTUSED 60
 15     INITTRANS 2
 16     STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```

SQL>
SQL>
SQL> INSERT INTO teaadmin.product
  2   (supplier_ref, region_ref, product_cat, product_des)
  3   SELECT supplier_ref, region_ref, product_cat, product_des
  4   FROM teaadmin.producttemp2;

```

1193 rows created.

```

SQL>
SQL>
SQL> -- VERIFY DATA
SQL> -- " Sample data Vendors by Product Report"
SQL>
SQL> select product_des, product_cat, supplier_ref, company_name
  2   FROM (
  3     select p.product_des, p.product_cat, p.supplier_ref, s.company_name
  4     from teaadmin.product p join teaadmin.supplier s
  5       on p.supplier_ref = s.supplier_ref
  6       and p.region_ref = s.region_ref
  7     ORDER BY DBMS_RANDOM.RANDOM)
  8 WHERE rownum < 11;

```

PRODUCT_DES	PRODUCT_CAT	SUPPLIER_REF	COMPANY_NAME
HOTEL REPS & CHAINS	350	11239	CROWNE PLAZA HOTEL GEORGIA
HOTEL REPS & CHAINS	350	11268	PLACE LOUIS RIEL ALL-SUITE HOTEL
TOUR OPERATORS/WHOLES	500	12302	GO ACTIVE VACATIONS
CRUISE LINES	310	1416	ALAMO RENT A CAR
CRUISE LINES	301	908	ELITE ORIENT TOURS INC.
TOUR OPERATORS/WHOLES	500	1182	GILL INT'L TRAVEL
MOTOR COACH TOUR OPER	400	8704	CARDINAL TRAVEL TOURS
ATTRACTI0NS	206	5081	ADVENTURE ISLAND TAMPA BAY, FLORIDA
AIRLINES	150	616	CATHAY PACIFIC AIRWAYS LTD
TOUR OPERATORS/WHOLES	500	12302	FRESH TRACKS CANADA

10 rows selected.

```

SQL> Prompt ****
*****
SQL> PROMPT Product create/load complete
Product create/load complete
SQL> PROMPT

```

```

SQL> PROMPT Press enter to finish Spool
Press enter to finish Spool
SQL> Prompt ****
*****

```

```
SQL> drop sequence teaadmin.adhoc_allsales_payment_ref_seq;
Sequence dropped.

SQL> ALTER TABLE adhoc_allsales drop column payment_ref;
Table altered.

SQL> create sequence teaadmin.adhoc_allsales_payment_ref_seq
  2  increment by 1
  3  nomaxvalue
  4  start with 999
  5  minvalue 1 ;
Sequence created.

SQL>
SQL>
SQL> ALTER TABLE adhoc_allsales add (payment_ref number(10));
Table altered.

SQL>
SQL> update teaadmin.adhoc_allsales set payment_ref = adhoc_allsales_payment_ref_seq.nextval;
1341 rows updated.
```

```

SQL> -----
SQL> -- ITINERARY HISTORY TABLE
SQL> -- Parent to Product
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.adhoc_allsales
SQL> --
SQL> -- TABLE NOTES - Using composite pk on itinerary, and payment for unique identifier on this table
SQL> -- -
SQL> --
SQL> --
SQL> -----
SQL>
SQL>
SQL>
SQL>
SQL> CREATE TABLE teaadmin.itinerary
  2     (itinerary_ref number(5),
  3      payment_ref number(5),
  4      booking_ref varchar2(10),
  5      supplier_ref number(5),
  6      region_ref number(1),
  7      product_cat number(5),
  8      product varchar2(30),
  9      prod_start date,
 10     prod_end date,
 11     destination_id varchar2(10),
 12     num_travellers number(2),
 13     travel_class varchar2(5),
 14     trip_type varchar2(15),
 15     itinerary_comments varchar2(70),
 16     CONSTRAINT payment_itin_pk PRIMARY KEY (payment_ref, itinerary_ref)
 17             USING INDEX (CREATE INDEX itin_inv_idx ON teaadmin.itinerary (payment_ref, itinerary_ref) PCTFREE 10
INITRANS 2
 18             STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
 19             TABLESPACE indx),
 20     CONSTRAINT itin_dest_id_fk FOREIGN KEY (destination_id) REFERENCES destination(dest_id) deferrable initially
immediate,
 21     CONSTRAINT itin_supp_reg_prod_id_fk FOREIGN KEY (supplier_ref, region_ref, product_cat) REFERENCES
product(supplier_ref, region_ref, product_cat) deferrable initially immediate)
 22     TABLESPACE COMPANYDATA
 23             PCTFREE 20
 24             PCTUSED 60
 25             INITTRANS 2
 26             STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```

SQL> -- Insert code violates FK but I don't know what is violating it.
SQL>
SQL> -- Code below violates the FK reference to the product PK.
SQL> -- This is because I cleaned the data in product as a separate table, and not in adhoc_directly.
SQL> -- Either redo everything or try and find a way around this...
SQL>
SQL> --INSERT INTO teaadmin.itinerary
SQL> --(itinerary_ref, payment_ref, booking_ref, supplier_ref, region_ref, product_cat,product, prod_start, prod_end,
destination_id,
SQL> -- num_travellers, travel_class, itinerary_comments)
SQL> --SELECT itinerary_num, payment_ref, booking_num, supplier_id, supp_office, product_cat, product, trip_start, trip_end,
dest_id,
SQL> --traveller_count,travelclass, product_des
SQL> --FROM teaadmin.adhoc_allsales;
SQL>
SQL> -- Error logging - what is causing this issue?
SQL>
SQL>
SQL>
SQL> begin
 2   dbms_errlog.create_error_log( dml_table_name => 'itinerary' );
 3 end;
 4 /

```

PL/SQL procedure successfully completed.

```

SQL>
SQL>
SQL> INSERT INTO teaadmin.itinerary
 2 (itinerary_ref, payment_ref, booking_ref, supplier_ref, region_ref, product_cat,product, prod_start, prod_end,
destination_id,
 3 num_travellers, travel_class, itinerary_comments)
 4 SELECT itinerary_num, payment_ref, booking_num, supplier_id, supp_office, product_cat, product, trip_start, trip_end,
dest_id,
 5 traveller_count,travelclass, product_des
 6 FROM teaadmin.adhoc_allsales
 7 log errors into ERR$ ITINERARY
 8 reject limit unlimited;

```

1341 rows created.

```

SQL>
SQL>
SQL> SELECT COUNT (*) from ITINERARY;

COUNT(*)
-----
1341

```

```

SQL>
SQL> Prompt ****
*****
SQL> PROMPT
SQL> PROMPT Enter to Continue
Enter to Continue
SQL> PROMPT

SQL> Prompt ****
*****
SQL> PAUSE

SQL>
SQL> -- 23 results originally... see commented out update code.
SQL>
SQL>
SQL>
SQL> --Check Error log
SQL> set pagesize 120
SQL> set linesize 130
SQL> col product_cat format a10
SQL> col supplier_ref format a10
SQL> col region_ref format a10
SQL> col product format a25
SQL> col ORA_ERR_MESG$ format a30
SQL>
SQL> select ORA_ERR_MESG$, supplier_ref, region_ref, product_cat, product from err$_itinerary
   2 order by supplier_ref, region_ref, product;

no rows selected

SQL>
SQL> /* TROUBLESHOOTING -
> -- Adding products and suppliers to their master lists. These lists were built with Ad_Hoc all sales and all suppliers
SQL> -- Both lists had some conflicting, unmatching info, so in order to make a master list complete we had to update some
-- more
SQL>
SQL> -- ORA-02291: integrity constraint (TEADMIN.ITIN_SUPP_REG_PROD_ID_FK) violated - parent key not found
SQL> -- No parent Product record?
SQL>
SQL> -- Insert into Prod
SQL>
SQL> INSERT INTO teaadmin.product (supplier_ref, region_ref, product_cat, product_des)
SQL> SELECT supplier_ref, region_ref, product_cat, product FROM ERR$_ITINERARY ;
SQL>

```

```

SQL> -- PRODUCT VIOLATION TOO? Not in official master supplier list?
SQL> -- Can add suppliers to list but will need to confirm all ID Details with TEA for updating.
SQL>
SQL> INSERT INTO teaadmin.supplier(supplier_ref, region_ref )
SQL> SELECT supplier_ref, region_ref FROM ERR$_ITINERARY ;
SQL>
SQL> --Violation of primary Key?
SQL> -- What Supplier is not in Master Supplier list - which is showing on the All Sales list?
SQL> -- Select comparison between error log suppliers and supplier
SQL>
SQL> col supp_ref format a5
SQL> col product_cat format 99999
SQL> col supplier_ref format 99999
SQL> col region_ref format 99999
SQL> col errsupp format A10
SQL> SELECT to_char(s.supplier_ref) supp_ref, s.region_ref, e.supplier_ref errsupp, s.region_ref suppreg
SQL> FROM supplier s RIGHT OUTER JOIN err$_itinerary e on s.supplier_ref = e.supplier_ref;
SQL>
SQL> col supplier_ref 99999
SQL> select supplier_ref from teaadmin.supplier where supplier_ref=150;
SQL>
SQL>
SQL> -- SALES HAS A ROGUE Supplier LISTINGS
SQL>
SQL> -- insert Rogue supplier to master supplier list.
SQL> INSERT INTO supplier (supplier_ref, region_ref, comments)
SQL> VALUES (150, 1, 'No record of Supplier in Supplier data, but shows on Sales list');
SQL>
SQL> INSERT INTO supplier (supplier_ref, region_ref, comments)
SQL> VALUES (2099, 2, 'No record of Supplier in Supplier data, but shows on Sales list');
SQL>
SQL> INSERT INTO supplier (supplier_ref, region_ref, comments)
SQL> VALUES (3600, 2, 'No record of Supplier in Supplier data, but shows on Sales list');
SQL>
SQL> -- Other issues Supplier, region, product voilation, not in product list. Will add,
SQL> INSERT INTO supplier (supplier_ref, region_ref, comments)
SQL> VALUES (3600, 0, 'No record of Supplier in Supplier data, but shows on Sales list');
SQL>
SQL> alter table product add (comments varchar2(50));
SQL> select supplier_ref, region_ref, product_cat from product where supplier_ref=150;
SQL> select supplier_ref, region_ref, product_cat from product where supplier_ref=100;
SQL> select supplier_ref, region_ref, product_cat from product where supplier_ref=2099;
SQL> select supplier_ref, region_ref, product_cat from product where supplier_ref=2099;
SQL>
SQL> INSERT INTO product (supplier_ref, region_ref, product_cat, comments)
SQL> VALUES (100,1,303, 'Updated information need to verify');
SQL> INSERT INTO product (supplier_ref, region_ref, product_cat, comments)
SQL> VALUES (150,1,150, 'Updated information need to verify');

```

```

SQL> INSERT INTO product (supplier_ref, region_ref, product_cat, comments)
SQL> VALUES (2099,1,102, 'Updated information need to verify');
SQL> INSERT INTO product (supplier_ref, region_ref, product_cat, comments)
SQL> VALUES (2099,2,100, 'Updated information need to verify');
SQL> INSERT INTO product (supplier_ref, region_ref, product_cat, comments)
SQL> VALUES (3600,2,502, 'Updated information need to verify');
SQL> INSERT INTO product (supplier_ref, region_ref, product_cat, comments)
SQL> VALUES (3600,2,500, 'Updated information need to verify');
SQL>
SQL>
SQL> */
SQL> -- error table
SQL>
SQL> set pagesize 120
SQL> set linesize 130
SQL> col product_cat format a10
SQL> col supplier_ref format a10
SQL> col region_ref format a10
SQL> col product format a25
SQL> col ORA_ERR_MESG$ format a30
SQL>
SQL> select ORA_ERR_MESG$, supplier_ref, region_ref, product_cat, product from err$_itinerary
   2 order by supplier_ref, region_ref, product;

no rows selected

SQL> -- All employee and customer information tables
SQL> -- are created, have data loaded, and constraints
SQL> -- enabled.
SQL> --
SQL> -- All supplier data and product information tables are
SQL> -- created, have data loaded and constraints enabled.
SQL> --
SQL> -- Most important! Data is clean!
SQL>
SQL> -----
SQL> Prompt ****
*****
SQL> PROMPT Proceed to Payment History, itinerary and Sales tables.
Proceed to Payment History, itinerary and Sales tables.
SQL> PROMPT

SQL> PROMPT
SQL> Prompt ****
*****
SQL> PAUSE

```

```

SQL>
SQL>
SQL> -----
SQL> -- PAYMENT HISTORY TABLE
SQL> -- Child to sale table
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.adhoc_allsales
SQL> --
SQL> -- SOURCE NOTES - Primary key cannot be itinerary_ref as it is not unique
SQL> --           - After some testing having payment_ref in payment table, I'm unable to load it into the
SQL> --           - itinerary table at the same time as other data from adhoc.
SQL> --           - Because of this, I'm adding the payment_ref to adhoc sales, it can be more efficiently used there.
SQL> --
SQL> --           - Watch for Alter Table FK reference after Fee Table creation
SQL> -----
SQL>
SQL>
SQL> --new
SQL> CREATE TABLE teaadmin.payment
  2  (payment_ref number(5),
  3  itinerary_ref number(5),
  4  creditcard_usetopay number(16),
  5  bill_date date,
  6  base_price number(10,2),
  7  fee_ref varchar2(5) ,
  8  fee_amt number(4),
  9  billed_amt number (10,2),
 10 total number (10,2),
 11 bill_des varchar2(50),
 12 CONSTRAINT payment_pk PRIMARY KEY (payment_ref) deferrable initially immediate USING INDEX
 13   (CREATE INDEX payment_idx ON teaadmin.payment(payment_ref)
 14     PCTFREE 10 INITTRANS 2
 15     STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
 16     TABLESPACE indx),
 17 CONSTRAINT pmt_fee_ref_fk FOREIGN KEY (fee_ref) REFERENCES fee(fee_ref) deferrable initially immediate)
 18 TABLESPACE COMPANYDATA
 19   PCTFREE 20
 20   PCTUSED 60
 21   INITTRANS 2
 22   STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```

SQL>
SQL>
```

```

SQL> INSERT INTO teaadmin.payment
  2  (payment_ref, itinerary_ref, creditcard_usetopay, bill_date, base_price, fee_ref, fee_amt, billed_amt, total, bill_des)
  3  SELECT payment_ref, itinerary_num, credit_card_num, bill_date, baseprice, agencyfee, feeamt, billedamt, totalpricewtax,
bill_des
  4  FROM teaadmin.adhoc_allsales;

1341 rows created.

SQL>
SQL> ---
SQL> drop sequence teaadmin.adhoc_allsales_comm_seq;
drop sequence teaadmin.adhoc_allsales_comm_seq
*
ERROR at line 1:
ORA-02289: sequence does not exist

SQL> ALTER TABLE adhoc_allsales drop column commission_ref;

Table altered.

SQL>
SQL> create sequence teaadmin.adhoc_allsales_comm_ref_seq
  2  increment by 1
  3  nomaxvalue
  4  start with 999
  5  minvalue 1 ;
create sequence teaadmin.adhoc_allsales_comm_ref_seq
*
Sequence created.

SQL>
SQL>
SQL> ALTER TABLE adhoc_allsales add (commission_ref number(10));

Table altered.

SQL>
SQL> update teaadmin.adhoc_allsales set commission_ref = adhoc_allsales_comm_ref_seq.nextval;

1341 rows updated.

```

```

SQL> -----
SQL> -- SALES TABLE
SQL> --     Bridge Table from Supplier to Itinerary to Customer to Employee
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.adhoc_allsales
SQL> --
SQL> -- SOURCE NOTES - commit works without the sale_supplier_code_fk after insert select below
SQL> --                         - will need to figure this out
SQL> --
SQL> -
SQL> -----
SQL>
SQL>
SQL>
SQL>
SQL> CREATE TABLE teaadmin.sale
  2     (supplier_ref number (5),
  3      region_ref number(1),
  4      itinerary_ref number(5),
  5      booking_ref varchar2(10),
  6      payment_ref number(5),
  7      cust_ref number(3),
  8      emp_ref number(5),
  9      agent_ref varchar2(2),
10      commission_ref number(10),
11      CONSTRAINT sale_pk PRIMARY KEY (payment_ref, itinerary_ref)
12          USING INDEX (CREATE INDEX sale_idx
13              ON teaadmin.sale (payment_ref, itinerary_ref)
14              PCTFREE 10 INITTRANS 2
15              STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
16              TABLESPACE indx),
17      CONSTRAINT sale_emp_fk FOREIGN KEY (emp_ref) REFERENCES employee (emp_ref) deferrable initially deferred,
18      CONSTRAINT sale_itinerary_fk FOREIGN KEY (payment_ref, itinerary_ref) REFERENCES itinerary (payment_ref,
itinerary_ref) deferrable initially immediate,
19      CONSTRAINT sale_payment_fk FOREIGN KEY (payment_ref) REFERENCES payment(payment_ref),
20      CONSTRAINT sale_cust_fk FOREIGN KEY (cust_ref) REFERENCES customer(cust_ref))
21          TABLESPACE userdata
22          PCTFREE 20
23          PCTUSED 60
24          INITTRANS 2
25          STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```

SQL>
SQL> COMMENT ON TABLE teaadmin.sale IS 'TEA sale Code Data';

```

Comment created.

```
SQL>
SQL> INSERT INTO teaadmin.sale (supplier_ref, region_ref, itinerary_ref, booking_ref, payment_ref, cust_ref, agent_ref,
commission_ref)
  2  SELECT supplier_id, supp_office, itinerary_num, booking_num, payment_ref, cust_id, agent, commission_ref
  3  FROM teaadmin.adhoc_allsales;
```

1341 rows created.

```
SQL>
SQL> commit;
```

Commit complete.

```
SQL>
SQL>
SQL> Desc SALE;
```

Name	Null?	Type
SUPPLIER_REF		NUMBER(5)
REGION_REF		NUMBER(1)
ITINERARY_REF	NOT NULL	NUMBER(5)
BOOKING_REF		VARCHAR2(10)
PAYMENT_REF	NOT NULL	NUMBER(5)
CUST_REF		NUMBER(3)
EMP_REF		NUMBER(5)
AGENT_REF		VARCHAR2(2)
COMMISSION_REF		NUMBER(10)

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```

SQL> -----
SQL> -- COMMISSION TRACKING TABLE
SQL> --
SQL> --
SQL> -- TABLESPACE - COMPANYDATA
SQL> -- SOURCE: teaadmin.adhoc_allsales
SQL> --
SQL> -- SOURCE NOTES - Lock down access to commission specialist only
SQL> --      -
SQL> --      -
SQL> -----
SQL>
SQL> -- temp table first, no constraints get sequence loaded
SQL>
SQL> CREATE TABLE teaadmin.commission
  2  (commission_ref number(5),
  3   supplier_ref number (5),
  4   region_ref number(1),
  5   expected_comm_pay date,
  6   booking_ref varchar2(10),
  7   agent varchar(2),
  8   commission_amount number (10,2),
  9   CONSTRAINT commission_pk PRIMARY KEY (commission_ref)
10      USING INDEX (CREATE INDEX commission_idx
11        ON teaadmin.commission (commission_ref)
12        PCTFREE 10 INITTRANS 2
13        STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
14        TABLESPACE indx))
15   TABLESPACE userdata
16      PCTFREE 20
17      PCTUSED 60
18      INITTRANS 2
19      STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);

```

Table created.

```

SQL>
SQL>
SQL>
SQL> INSERT INTO teaadmin.commission (commission_ref, supplier_ref, region_ref, expected_comm_pay, booking_ref,
commission_amount)
  2  SELECT commission_ref, supplier_id, supp_office, trip_end + 60, booking_num, agencycomm
  3  FROM teaadmin.adhoc_allsales;
1341 rows created.

SQL>
SQL> spool off

```

Verification

```
SQL> SET VERIFY ON
SQL> TTITLE OFF
SQL>
SQL>
SQL> -- Verification of database table creation and load database
SQL> -- Most (if not all?) tables had been verified after they were created in previous scripts,
SQL> -- In case we missed anything, we're checking again
SQL>

SQL> -- Describe the main tables
SQL>
SQL> SELECT * FROM tab;
```

TNAME	TABTYPE	CLUSTERID
ADHOC_ALLCLIENT	TABLE	
ADHOC_ALLSALES	TABLE	
ADHOC_ALLSALES2	TABLE	
ADHOC_ALLSUPPLIER	TABLE	
AFFILIATION	TABLE	
CC	TABLE	
COMMISSION	TABLE	
CUSTOMER	TABLE	
CUSTOMERCOMM	TABLE	
CUSTOMERDETAIL	TABLE	
DESTINATION	TABLE	
EMPLOYEE	TABLE	
EMPLOYEEDETAIL	TABLE	
FEE	TABLE	
ITINERARY	TABLE	
PAYMENT	TABLE	
PRODUCT	TABLE	
PRODUCTTEMP2	TABLE	
SALARY	TABLE	
SALE	TABLE	
SUPPLIER	TABLE	
SUPPLIERDETAIL	TABLE	
TRAVELCLASS	TABLE	

```
24 rows selected.
```

```

SQL> -- data counts
SQL>
SQL> DESC affiliation;
Name Null? Type
-----
AFFILIATION_CODE VARCHAR2(10)
AFFILIATION_DES VARCHAR2(50)

SQL>
SQL> SELECT count(*) from affiliation;

COUNT(*)
-----
7

SQL>
SQL> ttitle 'affiliation sample'
SQL> SELECT *
2   FROM  (SELECT * from affiliation
3          ORDER BY DBMS_RANDOM.RANDOM)
4 WHERE rownum < 5;

Sun Jan 08                               page    1
                                         affiliation sample

AFFILIATIO AFFILIATION_DES
-----
NEW
PGY
ACTA      Association of Canadian Travel Agents
ACTANEWP

SQL>
SQL>
SQL> DESC cc;
Name Null? Type
-----
CUST_REF NUMBER(3)
CREDITCARD NUMBER(16)
CC_TYPE VARCHAR2(10)
EXP_DATE DATE

SQL>

```

```
SQL> ttitle 'cc count'  
SQL> SELECT count(*) from cc;
```

Sun Jan 08

page 1

cc count

COUNT(*)
211

```
SQL> ttitle 'Credit Card sample'
```

```
SQL> set numw 16
```

```
SQL> SELECT *  
2      FROM  (SELECT * from cc  
3          ORDER BY DBMS_RANDOM.RANDOM)  
4     WHERE  rownum < 5;
```

Sun Jan 08

page 1

Credit Card sample

CUST_REF	CREDITCARD	CC_TYPE	EXP_DATE
380	28472397423987	Diners	05-MAY-16
248	7814724934239420	VISA	02-FEB-16
223	87632138542639	MC	01-JUL-14
142	632456487984533	AMEX	02-JAN-16

```
SQL>
```

```
SQL>
```

```
SQL> DESC commission;
```

Name	Null?	Type
COMMISSION_REF	NOT NULL	NUMBER(5)
SUPPLIER_REF		NUMBER(5)
REGION_REF		NUMBER(1)
EXPECTED_COMM_PAY		DATE
BOOKING_REF		VARCHAR2(10)
AGENT		VARCHAR2(2)
COMMISSION_AMOUNT		NUMBER(10,2)

```

SQL> ttitle 'Commission Sample'
SQL> SELECT count(*) from commission;

Sun Jan 08                               page      1
                                         Commission Sample

  COUNT(*)
-----
  1341

SQL>
SQL> ttitle 'comission sample'
SQL> SELECT *
  2   FROM  (SELECT commission_ref, supplier_ref, region_ref, booking_ref, expected_comm_pay from commission
  3         ORDER BY DBMS_RANDOM.RANDOM)
  4 WHERE  rownum < 5;

Sun Jan 08                               page      1
                                         Comission Sample

  COMMISSION_REF    SUPPLIER_REF    REGION_REF BOOKING_REF EXPECTED_
-----
  3471              717             1 5643658DF 23-AUG-14
  2641              3549            1 BGF5        25-JUL-13
  3356              3633            1 A6584768 18-AUG-14
  2439              9396            1 WDR901     28-JUN-16

SQL>
SQL> DESC customer;
      Name          Null?       Type
-----
CUST_REF           NUMBER(3)
FIRST_NAME         VARCHAR2(20)
LAST_NAME          VARCHAR2(20)
PREFERRED_AGENT   VARCHAR2(2)

SQL> ttitle 'Customer Count'
SQL> SELECT count(*) from customer;

Sun Jan 08                               page      1
                                         Customer Count

  COUNT(*)
-----
  270

```

```
SQL>
SQL> ttitle 'customer sample'
SQL> SELECT *
  2      FROM  (SELECT first_name, last_name, cust_ref from customer
  3              ORDER BY DBMS_RANDOM.RANDOM)
  4     WHERE rownum < 5;
```

Sun Jan 08 page 1
customer sample

FIRST NAME	LAST NAME	CUST REF
Bharat	Roberts	243
Paul	Intal	353
Sally	Naso	168
Kevin	Porch	272

SQL>

SQL>

```
SQL> DESC customerdetail;
```

Name	Null?	Type
CUST_REF		NUMBER(3)
HOME_PHONE		NUMBER(12)
BUSINESS_PHONE		NUMBER(12)
ADDRESS		VARCHAR2(35)
CITY		VARCHAR2(10)
PROV		VARCHAR2(2)
COUNTRY		VARCHAR2(10)
POSTALCODE		VARCHAR2(7)
EMAIL		VARCHAR2(25)
BIRTH_DATE		DATE
SIGNIFICANT_DATE		DATE
SIGNIFICANT_DATE_DESC		VARCHAR2(20)
REWARDS_NUMBER		NUMBER(15)
COMMENTS		VARCHAR2(50)

```
SQL> SELECT count(*) from customerdetail;
```

Sun Jan 08 page 1
customer sample

COUNT (*)

270

SQL>

```
SQL> ttitle 'customer detail sample'
SQL> SELECT *
  2    FROM  (SELECT cust_ref, home_phone, email from customerdetail
  3           ORDER BY DBMS_RANDOM.RANDOM)
  4  WHERE  rownum < 5;
```

```
Sun Jan 08                               page   1
                                              customer detail sample

  CUST REF      HOME PHONE EMAIL
-----  -----
  298        4032269966 pvalentine@aol.com
  110        4032693678 mmartin@nucleos.com
  180        4032959999 eatorres@telusplanet.net
  214        4035694566 ywilliams@home.com
```

```
SQL>
SQL>
SQL> DESC customercomm;
```

Name	Null?	Type
COMMUNICATION_REF		NUMBER(10)
CUST_REF		NUMBER(3)
LASTCONTACTDATE		DATE
LASTCONTACTTYPE		VARCHAR2(20)
EMP_REF		NUMBER(3)

```
SQL> ttitle 'Customer Communications'
SQL> SELECT count(*) from customercomm;
```

```
Sun Jan 08                               page   1
                                              Customer Communications

  COUNT(*)
-----
  26
```

```
SQL>
SQL> ttitle 'customer communication sample'
SQL> SELECT *
  2      FROM  (SELECT cust_ref, communication_ref, lastcontacttype, lastcontactdate from customercomm
  3            ORDER BY DBMS_RANDOM.RANDOM)
  4   WHERE  rownum < 5;
```

```
Sun Jan 08                               page    1
                                         customer communication sample

CUST_REF COMMUNICATION_REF LASTCONTACTTYPE      LASTCONTA
-----  -----  -----  -----
      109      1032 Email        01-NOV-15
      119      1040 Cold Call    15-OCT-15
      104      1027 Cold Call    15-OCT-15
      106      1029 Email        01-NOV-15
```

```
SQL>
SQL>
SQL> DESC destination;
Name          Null?    Type
-----  -----
DEST_ID        VARCHAR2(5)
DEST_DES       VARCHAR2(40)
```

```
SQL> title 'Destination'
SQL> SELECT count(*) from destination;
```

```
Sun Jan 08                               page    1
                                         Destination

COUNT(*)
-----
      10
```

```
SQL> ttitle 'destination sample'
SQL> SELECT *
      2      FROM  (SELECT * from destination
      3              ORDER BY DBMS_RANDOM.RANDOM)
      4     WHERE  rownum < 5;
```

Sun Jan 08 destination sample

DEST	DEST DES
MEAST	Middle East
SA	South America
OTHR	Other Destination
EU	Europe' & United Kingdom

```
SQL>
```

Name	Null?	Type
EMP_REF		NUMBER(3)
FIRST_NAME		VARCHAR2(20)
LAST_NAME		VARCHAR2(20)

```
SQL> SELECT count(*) from employee;
```

Sun Jan 08 Employee sample page 1

COUNT (*)

12

```
SQL> ttitle 'employee sample'
```

```
SQL> SELECT *
  2      FROM  (SELECT * from employee
  3              ORDER BY DBMS_RANDOM.RANDOM)
  4     WHERE rownum < 5;
```

Sun Jan 08 page 1
employee sample

EMP_REF	FIRST_NAME	LAST_NAME
110	Jolanda	W
107	Jane	Merrill
105	Bruce J.	Dixon
101	Judy	Lisle

SQL>

```
SQL> DESC employeedetail;
```

Name	Null?	Type
EMP_REF		NUMBER (3)
PREVAGENTID		VARCHAR2 (2)
HIRE_DATE		DATE
POSITION		VARCHAR2 (20)
ADDRESS		VARCHAR2 (40)
PHONE_NUMBER		NUMBER (10)
EMERGENCY_CONTACT		VARCHAR2 (10)
EMERGENCY_CONTACT_PHONE		NUMBER (10)
COMM		VARCHAR2 (50)

```
SQL> 'Emp Detail Sample'
```

```
SQL> SELECT count(*) from employeedetail;
```

```
Sun Jan 08
```

```
page 1
```

```
          Emp Detail Sample
```

COUNT (*)
12

```
SQL>
```

```
SQL> ttitle 'employee detail sample'
```

```
SQL> SELECT *
```

```
 2      FROM  (SELECT emp_ref, prevagentid, hire_date, position from employeedetail
 3            ORDER BY DBMS_RANDOM.RANDOM)
 4   WHERE  rownum < 5;
```

```
Sun Jan 08
```

```
page 1
```

```
          employee detail sample
```

EMP_REF	PR	HIRE_DATE	POSITION
---------	----	-----------	----------

109			
105	BD		Agent
107	JM		Agent
108	BP		Agent

```
SQL>
```

```
SQL> DESC fee;
```

Name	Null?	Type
FEE_REF		VARCHAR2 (5)
FEE_DES		VARCHAR2 (30)
AMOUNT		NUMBER (5,2)

```

SQL> ttitle 'fee sample'
SQL> SELECT count(*) from fee;
Sun Jan 08                               page      1
                                         fee sample

COUNT(*)
-----
7

SQL>
SQL> ttitle 'fee sample'
SQL>
SQL> SELECT *
  2   FROM  (SELECT * from fee
  3          ORDER BY DBMS_RANDOM.RANDOM)
  4 WHERE rownum < 5;

Sun Jan 08                               page      1
                                         fee sample

FEE_R FEE_DES                           AMOUNT
----- -----
GR    Group Booking                      100
RS    Research                           50
NSF   Insufficient Funds                25
CH    Change                            15

SQL>
SQL>
SQL> DESC itinerary;
      Name           Null?    Type
----- -----
ITINERARY_REF        NOT NULL NUMBER(5)
PAYMENT_REF          NOT NULL NUMBER(5)
BOOKING_REF          VARCHAR2(10)
SUPPLIER_REF          NUMBER(5)
REGION_REF            NUMBER(1)
PRODUCT_CAT           NUMBER(5)
PRODUCT               VARCHAR2(30)
PROD_START             DATE
PROD_END               DATE
DESTINATION_ID        VARCHAR2(10)
NUM_TRAVELLERS        NUMBER(2)
TRAVEL_CLASS           VARCHAR2(5)
TRIP_TYPE              VARCHAR2(15)
ITINERARY_COMMENTS    VARCHAR2(70)

```

```

SQL> ttitle 'itinerary sample'
SQL> SELECT count(*) from itinerary;

Sun Jan 08                               page      1
                                         itinerary sample

  COUNT(*)
-----
  1341

SQL>
SQL> ttitle 'itinerary sample'
SQL> set linesize 100
SQL> SELECT *
  2   FROM  (SELECT itinerary_ref, booking_ref, product, prod_start, prod_end from itinerary
  3          ORDER BY DBMS_RANDOM.RANDOM)
  4 WHERE  rownum < 5;

Sun Jan 08                               page      1
                                         itinerary sample

  ITINERARY_REF BOOKING_REF PRODUCT           PROD_STAR PROD_END
-----  -----
    428 FSD82945   Airlines        08-JAN-15 09-JAN-15
    862 659874     Travel Insurance 30-MAY-14 24-JUN-14
    185 KK890       Tour Operators/Wholes 01-APR-16 25-APR-16
    299 TRE783      Railroads      14-MAY-15 08-JUN-15

SQL>
SQL>
SQL> DESC payment;
      Name          Null?    Type
-----  -----
PAYMENT_REF          NUMBER(5)
ITINERARY_REF        NUMBER(5)
BILL_DATE            DATE
BASE_PRICE           NUMBER(10,2)
FEE_REF              VARCHAR2(5)
FEE_AMT              NUMBER(4)
BILLED_AMT          NUMBER(10,2)
TOTAL                NUMBER(10,2)
BILL_DES             VARCHAR2(50)

SQL>

```

```

SQL> ttitle 'payment sample'
SQL> SELECT count(*) from payment;
Sun Jan 08                               page      1
                                         payment sample

  COUNT(*)
-----
  1341

SQL>
SQL> ttitle 'payment sample'
SQL> SELECT *
  2   FROM  (SELECT payment_ref, itinerary_ref, bill_date from payment
  3          ORDER BY DBMS_RANDOM.RANDOM)
  4 WHERE rownum < 5;

Sun Jan 08                               page      1
                                         payment sample

  PAYMENT REF  ITINERARY REF BILL DATE
-----  -----
  2117          862 28-APR-14
  1608          597 20-DEC-13
  1289          914 16-APR-13
  1612          629 18-JAN-14

SQL>
SQL> DESC product;
      Name           Null?    Type
-----  -----
SUPPLIER_REF           NOT NULL NUMBER(5)
REGION_REF            NOT NULL NUMBER(5)
PRODUCT_CAT           NOT NULL NUMBER(5)
PRODUCT_DES           VARCHAR2(30)
COMMENTS              VARCHAR2(50)

SQL> ttitle 'product sample'
SQL> SELECT count(*) from product;
Sun Jan 08                               page      1
                                         product sample

  COUNT(*)
-----
  1199

```

```

SQL> SELECT *
2      FROM  (SELECT supplier_ref, region_ref, product_cat, product_des from product
3            ORDER BY DBMS_RANDOM.RANDOM)
4   WHERE  rownum < 5;

```

Sun Jan 08

page 1

product sample

SUPPLIER REF	REGION REF	PRODUCT CAT	PRODUCT DES
3119	1	506	TOUR OPERATORS/WHOLES
3562	1	600	YACHT & BOAT CHARTERS
9285	1	500	TOUR OPERATORS/WHOLES
12854	1	500	TOUR OPERATORS/WHOLES

SQL>

SQL>

SQL> DESC salary;

Name	Null?	Type
EMP_REF		NUMBER(3)
LASTREVIEWDATE		DATE
CURRMONTHSAL	NOT NULL	NUMBER(7,2)
PREVMONTHSAL		NUMBER(7,2)
INCREASEREASON		VARCHAR2(70)

SQL> ttitle 'salary sample'

SQL> SELECT count(*) from salary;

Sun Jan 08

page 1

salary sample

COUNT(*)
12

```

SQL>
SQL> COL increasereason format a15
SQL> ttitle 'salary sample'
SQL> SELECT *
  2     FROM  (SELECT emp_ref, lastreviewdate , currmonthsal, increasereason from salary
  3           ORDER BY DBMS_RANDOM.RANDOM)
  4      WHERE rownum < 5;

```

Sun Jan 08

salary sample			
EMP_REF	LASTREVIE	CURRMONTHSAL	INCREASEREASON
106		800	
102	09-OCT-15	1400	Satisfactory Review. Pay Increase.
110	11-OCT-15	1700	Satisfactory Review. Pay Increase.
101	09-OCT-15	1400	Satisfactory Review. Pay Increase.

page 1

```

SQL>
SQL>
SQL> DESC sale;
Name          Null?    Type
-----          -----
SUPPLIER_REF          NUMBER(5)
REGION_REF          NUMBER(1)
ITINERARY_REF        NOT NULL NUMBER(5)
BOOKING_REF          VARCHAR2(10)
PAYMENT_REF          NOT NULL NUMBER(5)
CUST_REF             NUMBER(3)
EMP_REF              NUMBER(5)
AGENT_REF             VARCHAR2(2)
COMMISSION_REF        NUMBER(10)

```

```

SQL> ttitle 'sale sample'
SQL> SELECT count(*) from sale;

Sun Jan 08                               page    1
                                         sale sample

  COUNT(*)
-----
  1341

SQL>
SQL> SELECT *
  2   FROM  (SELECT supplier_ref, region_ref ,itinerary_ref, booking_ref from sale
  3          ORDER BY DBMS_RANDOM.RANDOM)
  4  WHERE  rownum < 5;

Sun Jan 08                               page    1
                                         sale sample

SUPPLIER_REF      REGION_REF      ITINERARY_REF BOOKING_REF
-----           -----           -----           -----
  3212             1               285 GFF77
  420              1               308 MKI336
  3600             2               1106 AD7889
  3212             1               518 SG4SD

SQL>
SQL>
SQL> DESC supplier;
      Name                Null?    Type
-----
SUPPLIER_REF
REGION_REF
REPRESENTITIVE_CODE
COMPANY_NAME
COMMENTS

```

```
SQL> ttitle 'supplier sample'  
SQL> SELECT count(*) from supplier;
```

Sun Jan 08

page 1

supplier sample

COUNT(*)

735

```
SQL>  
SQL>  
SQL>  
SQL> SELECT *  
2      FROM  (SELECT supplier_ref, region_ref , company_name from supplier  
3          ORDER BY DBMS RANDOM.RANDOM)  
4      WHERE  rownum < 5;
```

Sun Jan 08

page 1

supplier sample

SUPPLIER_REF	REGION_REF	COMPANY_NAME
2623	1	SANDALS AND BEACHES RESORTS
3728	1	ARION TRAVEL
12278	1	PLATINUM HOSPITALITY GROUP
11692	1	EMIRATES

```
SQL>  
SQL> DESC supplierdetail;
```

Name	Null?	Type
SUPPLIER_REF		NUMBER (5)
REGION_REF		NUMBER(1)
PHONE_NUMBER		NUMBER(12)
FAX_NUMBER		NUMBER(12)
ADDRESS		VARCHAR2(35)
ADDRESS2		VARCHAR2(30)
CITY		VARCHAR2(25)
PROV		VARCHAR2(2)
POSTAL_CODE		VARCHAR2(7)
COUNTRY		VARCHAR2(20)
CONTACT_NAME		VARCHAR2(25)
EMAIL		VARCHAR2(45)
WEB		VARCHAR2(40)
AFFILIATION_CODE		VARCHAR2(15)
COMMENTS		VARCHAR2(100)

```

SQL>
SQL> SELECT count(*) from supplierdetail;
Sun Jan 08                                page      1
                                         supplier sample

  COUNT(*)
-----
  731

SQL>
SQL> ttitle 'supplierdetail sample'
SQL>
SQL> SELECT *
  2   FROM  (SELECT supplier_ref, region_ref , web from supplierdetail WHERE web is not null
  3          ORDER BY DBMS_RANDOM.RANDOM)
  4 WHERE  rownum < 5;

Sun Jan 08                                page      1
                                         supplierdetail sample

  SUPPLIER_REF      REGION_REF WEB
-----
  3015                  1 http://www.median-aviation.com
  742                   1 http://www.contiki.com
  1448                  1 http://www.hojo-canada.com
  119                   1 http://www.albatours.com

SQL>
SQL>
SQL> DESC travelclass;
      Name                Null?    Type
-----
  TRAVEL_CLASS            VARCHAR2(5)
  CLASS_DES              VARCHAR2(20)

SQL>
SQL> SELECT count(*) from travelclass;
Sun Jan 08                                page      1

  COUNT(*)
-----
  8

```

```

SQL>
SQL> ttitle 'travel class sample'
SQL> SELECT *
  2      FROM  (SELECT * from travelclass
  3          ORDER BY DBMS_RANDOM.RANDOM)
  4      WHERE  rownum < 5;

Sun Jan 08                               page     1
                                         travel class sample

TRAVE CLASS DES
-----
OCNV Ocean View
DBL Double
BSN Business
ECN Economy

SQL> -- Sample Sales Report
SQL> -- Join on Customer, Sale, Itinerary, Product, and Payment
SQL>
SQL> SET linesize 130
SQL>
SQL> col product_cat format 999
SQL> col first_name format a10
SQL> col last_name format a10
SQL> TTITLE 'Sample Yacht Package Sales from 2015'
SQL> SELECT *
  2      FROM  (
  3          SELECT distinct c.first_name, c.last_name, s.booking_ref, i.product_cat, p.product_des,
pa.bill_date, pa.billed_amt, pa.total
  4          FROM customer c JOIN sale s ON c.cust_ref = s.cust_ref JOIN itinerary i ON s.itinerary_ref =
i.itinerary_ref JOIN product p ON i.supplier_ref = p.supplier_ref
  5          JOIN payment pa ON s.payment_ref = pa.payment_ref
  6          WHERE pa.bill_date >= to_date('31 Dec 2014', 'DD MON YYYY') AND pa.bill_date <= to_date('1 Jan
2016', 'DD MON YYYY')

```

```
7      AND p.PRODUCT_DES LIKE 'YACHT%'
8      ORDER BY DBMS_RANDOM.RANDOM)
9      WHERE rownum < 5;
```

Sun Jan 08

page 1

Sample Yacht Package Sales from 2015

FIRST NAME	LAST NAME	BOOKING RE	PRODUCT CAT	PRODUCT DES	BILL DATE	BILLED AMT	TOTAL
Jonathan	Jaele	SDFD898	608	YACHT & BOAT CHARTERS	01-APR-15	803	1605
Eric	Da Silva	SDF45	602	YACHT & BOAT CHARTERS	21-APR-15	1070	2140
Eric	Da Silva	SDF45	602	YACHT & BOAT CHARTERS	27-MAR-15	1170	2140
Jonathan	Jaele	SDFD898	608	YACHT & BOAT CHARTERS	16-MAR-15	903	1605

```
SQL>
SQL> ttitle off
SQL>
SQL>
SQL>
SQL>
SQL> -- data verification good
SQL>
```

```

SQL> -- constraint check
SQL>
SQL> TTITLE 'Constraint checks'
SQL> col table_name format A15
SQL> col constraint_name format A20
SQL> col constraint_type format A5 heading 'type'
SQL> col status format A15
SQL> col owner format A15
SQL> set pagesize 50
SQL> set linesize 80
SQL> SELECT table_name, constraint_name, constraint_type, status, owner
2   FROM user_constraints;

```

Sun Jan 08

page 1

Constraint checks

TABLE NAME	CONSTRAINT NAME	type	STATUS	OWNER
CUSTOMERCOMM	CUSTCOMMREF PK	P	ENABLED	TEAADMIN
CUSTOMERCOMM	COMM_CUST_REF_FK	R	ENABLED	TEAADMIN
AFFILIATION	AFFILIATION_CODE_PK	P	ENABLED	TEAADMIN
EMPLOYEE	EMPREF PK	P	ENABLED	TEAADMIN
EMPLOYEEDETAIL	EMP_PK	P	ENABLED	TEAADMIN
EMPLOYEEDETAIL	EMPDET_EMP FK	R	DISABLED	TEAADMIN
SUPPLIER	SUPPLIER_REGION_PK	P	ENABLED	TEAADMIN
SUPPLIERDETAIL	DETAILS_SUPP_REGION_ PK	P	ENABLED	TEAADMIN
SALARY	SAL_NN	C	ENABLED	TEAADMIN
SALARY	SALARY PK	P	ENABLED	TEAADMIN
SALARY	EMPSAL_EMP_FK	R	ENABLED	TEAADMIN
SUPPLIERDETAIL	AFFIL_SUP_DETAIL_FK	R	ENABLED	TEAADMIN
SUPPLIERDETAIL	SUPPDETAIL_FK	R	ENABLED	TEAADMIN
CUSTOMER	CUST_REF PK	P	ENABLED	TEAADMIN
CUSTOMERDETAIL	CUST_DET_REF_PK	P	ENABLED	TEAADMIN
CUSTOMERDETAIL	CUST_DET_FK	R	ENABLED	TEAADMIN
PRODUCT	PROD_SUPP_REG_PK	P	ENABLED	TEAADMIN
CC	CC_PK	P	ENABLED	TEAADMIN
CC	CC_CUST_FK	R	ENABLED	TEAADMIN
ITINERARY	PAYMENT_ITIN PK	P	ENABLED	TEAADMIN
PRODUCT	SUPP_FK	R	ENABLED	TEAADMIN
ITINERARY	ITIN_DEST_ID_FK	R	ENABLED	TEAADMIN

ITINERARY	ITIN_SUPP_REG_PROD_I D_FK	R	ENABLED	TEAADMIN
PAYMENT	PAYMENT_PK	P	ENABLED	TEAADMIN
PAYMENT	PMT_FEE_REF_FK	R	ENABLED	TEAADMIN
SALE	SALE_PK	P	ENABLED	TEAADMIN
SALE	SALE_EMP_FK	R	ENABLED	TEAADMIN
SALE	SALE_ITINERARY_FK	R	ENABLED	TEAADMIN
FEE	FEE_REF_PK	P	ENABLED	TEAADMIN
DESTINATION	DEST_PK	P	ENABLED	TEAADMIN
TRAVELCLASS	TRAVEL_CLASS_PK	P	ENABLED	TEAADMIN
SALE	SALE_PAYMENT_FK	R	ENABLED	TEAADMIN
SALE	SALE_CUST_FK	R	ENABLED	TEAADMIN
COMMISSION	COMMISSION_PK	P	ENABLED	TEAADMIN

34 rows selected.

```

SQL>
SQL>    TTITLE off
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> spool off

```

SQL LDR and Control Files

adhocclientcontrol.ctl

```
--load all client data to adhoc with this control file

--sqlldr teaadmin/teaadmin control=C:\tea\csv\control\adhocclientcontrol.ctl log=C:\tea\csv\control\controllogs\adhocclientcontrol.log
bad=C:\tea\csv\control\controllogs\badadhocclient.log

--controlfile for sqlldr
options(SKIP=1, direct=y)
LOAD DATA
  INFILE 'C:\TEA\csv\clientdata.csv'
    TRUNCATE INTO TABLE adhoc_allclient
    FIELDS TERMINATED BY "," optionally enclosed by ""
TRAILING NULLCOLS
  (cust_id integer external,
  first_name char,
  last_name char,
  agent char,
  email char,
  home_phone integer external,
  business_phone integer external,
  birthdate DATE 'DD-MM-YY',
  address char,
  city char,
  postal_code char,
  prov char,
  country char,
  comments char)
```

adhocsalescontrol.ctl

```
--controlfile for sqlldr ALL SALES DATA
--sqlldr teaadmin/teaadmin control=C:\tea\csv\control\adhocsalescontrol.ctl log=C:\tea\csv\control\controllogs\adhocsalescontrol.log
bad=C:\tea\csv\control\controllogs\badadhocsales.log

options(SKIP=1, direct=y)
LOAD DATA
  INFILE 'C:\TEA\csv\sales.csv'
    truncate INTO TABLE adhoc_allsales
    FIELDS TERMINATED BY "," optionally enclosed by ""
TRAILING NULLCOLS
  (sale_date DATE 'DD/MM/YYYY',
  cust_id integer external,
  itinerary_num integer external,
  agent char,
  booking_num char,
  product_cat integer external,
  supplier_id integer external,
```

```
Supp_office integer external,
trip_start DATE 'DD-MM-YYYY',
trip_end DATE 'DD/MM/YYYY',
travelclass char,
traveller count integer external,
product char,
product_des char,
destination char,
dest_id char,
credit_card char,
credit_card_exp DATE 'DD/MM/YYYY',
credit_card_num integer external,
bill_date DATE 'DD/MM/YYYY',
bill_des char,
baseprice integer external,
totalpricewtax integer external,
billedamt integer external,
agencyfee integer external,
feeamt integer external,
agencycomm integer external)
```

adhocsuppliercontrol.ctl

```
--sqlldr teaadmin/teaadmin control=C:\tea\csv\control\adhocsuppliercontrol.ctl log=C:\tea\csv\control\controllogs\adhocsuppliercontrol.log
bad=C:\tea\csv\control\controllogs\badadhocsupplier.log

--controlfile for sqlldr
options(SKIP=1, direct=y)
LOAD DATA
  INFILE 'C:\TEA\csv\suppdata.csv'
    TRUNCATE INTO TABLE adhoc_allsupplier
    FIELDS TERMINATED BY "," optionally enclosed by ""
    TRAILING NULLCOLS
(supplier_id integer external,
product_cat integer external,
Supp_office integer external,
Prod_Des char,
Contact char,
company char,
Supp_Address char,
Supp_Address2 char,
city char,
prov char,
postalcode char,
country char,
Phone integer external,
Fax integer external,
email char,
web char,
representative char,
affiliation char,
comments char)
```

```
affiliation.ctl
```

```
--options(SKIP=1, direct=y)
--
--c:\sqlldr teaadmin/teaadmin control=C:\tea\csv\control\affiliation.ctl log=C:\tea\csv\control\controllogs\affiliation.log
bad=C:\tea\csv\control\controllogs\badaffiliation.log

options(skip=1, direct=y)
LOAD DATA
infile 'C:\TEA\csv\Affiliation.csv'
--INFILE *
TRUNCATE INTO TABLE teaadmin.affiliation
FIELDS TERMINATED BY "," optionally enclosed by ""
TRAILING NULLCOLS
(affiliation_code char,
affiliation_des char)
```

```
affiliation.log
```

```
SQL*Loader: Release 11.2.0.1.0 - Production on Sun Jan 8 11:07:38 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Control File: C:\tea\csv\control\affiliation.ctl
Data File: C:\TEA\csv\Affiliation.csv
Bad File: C:\tea\csv\control\controllogs\badaffiliation.log
Discard File: none specified

(Allow all discards)

Number to load: ALL
Number to skip: 1
Errors allowed: 50
Continuation: none specified
Path used: Direct

Table TEAADMIN.AFFILIATION, loaded from every logical record.
Insert option in effect for this table: TRUNCATE
TRAILING NULLCOLS option in effect

      Column Name          Position    Len   Term Encl Datatype
-----+-----+-----+-----+-----+-----+
AFFILIATION_CODE           FIRST      *     ,  O(") CHARACTER
AFFILIATION_DES            NEXT      *     ,  O(") CHARACTER

The following index(es) on table TEAADMIN.AFFILIATION were processed:
index TEAADMIN.AFFILIATION_IDX loaded successfully with 7 keys

Table TEAADMIN.AFFILIATION:
 7 Rows successfully loaded.
 0 Rows not loaded due to data errors.
 0 Rows not loaded because all WHEN clauses were failed.
```

```
0 Rows not loaded because all fields were null.

Bind array size not used in direct path.
Column array rows :      5000
Stream buffer bytes:  256000
Read    buffer bytes: 1048576

Total logical records skipped:          1
Total logical records read:           7
Total logical records rejected:        0
Total logical records discarded:       0
Total stream buffers loaded by SQL*Loader main thread:     1
Total stream buffers loaded by SQL*Loader load thread:    0

Run began on Sun Jan 08 11:07:38 2017
Run ended on Sun Jan 08 11:07:53 2017

Elapsed time was:      00:00:15.25
CPU time was:          00:00:00.05
```

classtypecontrol.ctl

```
--load all client data to adhoc with this control file
--sqlldr teaadmin/teaadmin control=C:\tea\csv\control\classtypecontrol.ctl log=C:\tea\csv\control\controllogs\classtype_control.log
bad=C:\tea\csv\control\controllogs\badclasstype.log

--controlfile for sqlldr
options(SKIP=1, direct=y)
LOAD DATA
  INFILE 'C:\TEA\csv\classtype.csv'
  TRUNCATE INTO TABLE teaadmin.travelclass
  FIELDS TERMINATED BY "," optionally enclosed by ""
  TRAILING NULLCOLS
  (travel_class char,
   class Des char)
```

classtype_control.log

```
SQL*Loader: Release 11.2.0.1.0 - Production on Sun Jan 8 10:51:43 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Control File:  C:\tea\csv\control\classtypecontrol.ctl
Data File:    C:\TEA\csv\classtype.csv
Bad File:    C:\tea\csv\control\controllogs\badclasstype.log
Discard File: none specified

(Allow all discards)

Number to load: ALL
Number to skip: 1
Errors allowed: 50
Continuation:  none specified
```

```

Path used: Direct

Table TEAADMIN.TRAVELCLASS, loaded from every logical record.
Insert option in effect for this table: TRUNCATE
TRAILING NULLCOLS option in effect

Column Name          Position  Len  Term Encl Datatype
-----              -----   --  --  --  --
TRAVEL_CLASS           FIRST     * , O(") CHARACTER
CLASS_DES             NEXT     * , O(") CHARACTER

The following index(es) on table TEAADMIN.TRAVELCLASS were processed:
index TEAADMIN.TRAVEL_CLASS_IDX loaded successfully with 8 keys

Table TEAADMIN.TRAVELCLASS:
8 Rows successfully loaded.
0 Rows not loaded due to data errors.
0 Rows not loaded because all WHEN clauses were failed.
0 Rows not loaded because all fields were null.

Bind array size not used in direct path.
Column array rows : 5000
Stream buffer bytes: 256000
Read    buffer bytes: 1048576

Total logical records skipped: 1
Total logical records read: 8
Total logical records rejected: 0
Total logical records discarded: 0
Total stream buffers loaded by SQL*Loader main thread: 1
Total stream buffers loaded by SQL*Loader load thread: 0

Run began on Sun Jan 08 10:51:43 2017
Run ended on Sun Jan 08 10:51:58 2017

Elapsed time was: 00:00:15.27
CPU time was: 00:00:00.03

```

customercomm.ctl

```

--options(SKIP=1, direct=y)
--
--c:\
--sqlldr teaadmin/teaadmin control=C:\tea\csv\control\customercomm.ctl log=C:\tea\csv\control\controllogs\customercomm.log
bad=C:\tea\csv\control\controllogs\badcustcomm.log
options(skip=1, direct=y)
LOAD DATA
infile 'C:\TEA\csv\customercomm.csv'
--INFILE *
TRUNCATE INTO TABLE teaadmin.customercommtemp
FIELDS TERMINATED BY "," optionally enclosed by "'"

```

```

TRAILING NULLCOLS
(communication_ref filler,
cust_ref integer external,
lastcontactdate date 'MM/DD/YYYY',
lastcontacttype char,
emp_ref integer external)

```

customercomm.log

```

SQL*Loader: Release 11.2.0.1.0 - Production on Sat Jan 7 18:53:06 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Control File: C:\tea\csv\control\customercomm.ctl
Data File: C:\TEA\csv\customercomm.csv
Bad File: C:\tea\csv\control\controllogs\badcustcomm.log
Discard File: none specified

(Allow all discards)

Number to load: ALL
Number to skip: 1
Errors allowed: 50
Continuation: none specified
Path used: Direct

Table TEADMIN.CUSTOMERCOMMTEMP, loaded from every logical record.
Insert option in effect for this table: TRUNCATE
TRAILING NULLCOLS option in effect

      Column Name          Position   Len   Term Encl Datatype
-----  -----  -----  -----  -----  -----
COMMUNICATION_REF           FIRST     * , O(") CHARACTER
(FILLER FIELD)
CUST_REF                   NEXT      * , O(") CHARACTER
LASTCONTACTDATE            NEXT      * , O(") DATE MM/DD/YYYY
LASTCONTACTTYPE             NEXT      * , O(") CHARACTER
EMP_REF                     NEXT      * , O(") CHARACTER

Table TEADMIN.CUSTOMERCOMMTEMP:
26 Rows successfully loaded.
0 Rows not loaded due to data errors.
0 Rows not loaded because all WHEN clauses were failed.
0 Rows not loaded because all fields were null.

Date cache:
Max Size:      1000
Entries :        2
Hits :         24
Misses :       0

Bind array size not used in direct path.

```

```

Column array rows : 5000
Stream buffer bytes: 256000
Read buffer bytes: 1048576

Total logical records skipped: 1
Total logical records read: 26
Total logical records rejected: 0
Total logical records discarded: 0
Total stream buffers loaded by SQL*Loader main thread: 1
Total stream buffers loaded by SQL*Loader load thread: 0

Run began on Sat Jan 07 18:53:06 2017
Run ended on Sat Jan 07 18:53:06 2017

Elapsed time was: 00:00:00.17
CPU time was: 00:00:00.02

```

employeecontrol.ctl

```

--options(SKIP=1, direct=y)
--
--c:\
--sqlldr teaadmin/teaadmin control=C:\tea\csv\control\employeecontrol.ctl log=C:\tea\csv\control\controllogs\empcontrol.log
bad=C:\tea\csv\control\controllogs\bademp.log

options(skip=1, direct=y)
LOAD DATA
infile 'C:\TEA\csv\employee.csv'
--INFILE *
TRUNCATE INTO TABLE teaadmin.employee
  FIELDS TERMINATED BY "," optionally enclosed by """
  TRAILING NULLCOLS
  (emp_ref integer external,
  first_name char,
  last_name char)

```

empcontrol.log

```

SQL*Loader: Release 11.2.0.1.0 - Production on Sat Jan 7 18:48:58 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Control File: C:\tea\csv\control\employeecontrol.ctl
Data File: C:\TEA\csv\employee.csv
Bad File: C:\tea\csv\control\controllogs\bademp.log
Discard File: none specified

(Allow all discards)

Number to load: ALL
Number to skip: 1
Errors allowed: 50
Continuation: none specified

```

```

Path used: Direct

Table TEAADMIN.EMPLOYEE, loaded from every logical record.
Insert option in effect for this table: TRUNCATE
TRAILING NULLCOLS option in effect

Column Name          Position   Len   Term Encl Datatype
-----              -----      --  --  --  --
EMP_REF             FIRST      * ,  O(") CHARACTER
FIRST_NAME           NEXT      * ,  O(") CHARACTER
LAST_NAME            NEXT      * ,  O(") CHARACTER

The following index(es) on table TEAADMIN.EMPLOYEE were processed:
index TEAADMIN.EMPREF_IDX loaded successfully with 12 keys

Table TEAADMIN.EMPLOYEE:
12 Rows successfully loaded.
0 Rows not loaded due to data errors.
0 Rows not loaded because all WHEN clauses were failed.
0 Rows not loaded because all fields were null.

Bind array size not used in direct path.
Column array rows : 5000
Stream buffer bytes: 256000
Read    buffer bytes: 1048576

Total logical records skipped: 1
Total logical records read: 12
Total logical records rejected: 0
Total logical records discarded: 0
Total stream buffers loaded by SQL*Loader main thread: 1
Total stream buffers loaded by SQL*Loader load thread: 0

Run began on Sat Jan 07 18:48:58 2017
Run ended on Sat Jan 07 18:49:13 2017

Elapsed time was: 00:00:15.23
CPU time was: 00:00:00.08

```

employeedetailcontrol.ctl

```

--options(SKIP=1, direct=y)
--c:\
--sqlldr teaadmin/teaadmin control=C:\tea\csv\control\employeedetailcontrol.ctl log=C:\tea\csv\control\controllogs\empdetailcontrol.log
bad=C:\tea\csv\control\controllogs\badempdetail.log

options(skip=1, direct=y)
LOAD DATA
infile 'C:\TEA\csv\employeedetail.csv'
--INFILE *
TRUNCATE INTO TABLE teaadmin.employeedetail
FIELDS TERMINATED BY "," optionally enclosed by ""
TRAILING NULLCOLS
(emp ref integer external,

```

```
prevagentid char,
hire_date filler,
position char,
address char,
phone_number integer external,
emergency_contact char,
emergency_contact_phone integer external,
comm char)
```

empdetailcontrol.log

```
SQL*Loader: Release 11.2.0.1.0 - Production on Sat Jan 7 18:50:46 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Control File: C:\tea\csv\control\employeedetailcontrol.ctl
Data File: C:\TEA\csv\employeedetail.csv
Bad File: C:\tea\csv\control\controllogs\badempdetail.log
Discard File: none specified

(Allow all discards)
```

```
Number to load: ALL
Number to skip: 1
Errors allowed: 50
Continuation: none specified
Path used: Direct
```

```
Table TEAADMIN.EMPLOYEEDETAIL, loaded from every logical record.
Insert option in effect for this table: TRUNCATE
TRAILING NULLCOLS option in effect
```

Column Name	Position	Len	Term	Encl	Datatype
EMP_REF	FIRST	*	,	O(")	CHARACTER
PREVAGENTID	NEXT	*	,	O(")	CHARACTER
HIRE_DATE (FILLER FIELD)	NEXT	*	,	O(")	CHARACTER
POSITION	NEXT	*	,	O(")	CHARACTER
ADDRESS	NEXT	*	,	O(")	CHARACTER
PHONE_NUMBER	NEXT	*	,	O(")	CHARACTER
EMERGENCY_CONTACT	NEXT	*	,	O(")	CHARACTER
EMERGENCY_CONTACT_PHONE	NEXT	*	,	O(")	CHARACTER
COMM	NEXT	*	,	O(")	CHARACTER

```
Referential Integrity Constraint/Trigger Information:
NULL, UNIQUE, and PRIMARY KEY constraints are unaffected.
```

```
Constraint TEAADMIN.EMPLOYEEDETAIL.EMPDET_EMP_FK was disabled and novalidated before the load.
The following index(es) on table TEAADMIN.EMPLOYEEDETAIL were processed:
index TEAADMIN.EMP_IDX loaded successfully with 12 keys
```

```
Table TEAADMIN.EMPLOYEEDETAIL has no constraint exception table.  
No CHECK, REFERENTIAL constraints were re-enabled after the load.
```

```
Table TEAADMIN.EMPLOYEEDETAIL:
```

```
 12 Rows successfully loaded.  
 0 Rows not loaded due to data errors.  
 0 Rows not loaded because all WHEN clauses were failed.  
 0 Rows not loaded because all fields were null.
```

```
Bind array size not used in direct path.
```

```
Column array rows : 5000  
Stream buffer bytes: 256000  
Read   buffer bytes: 1048576
```

```
Total logical records skipped: 1  
Total logical records read: 12  
Total logical records rejected: 0  
Total logical records discarded: 0  
Total stream buffers loaded by SQL*Loader main thread: 1  
Total stream buffers loaded by SQL*Loader load thread: 0
```

```
Run began on Sat Jan 07 18:50:46 2017
```

```
Run ended on Sat Jan 07 18:51:02 2017
```

```
Elapsed time was: 00:00:15.54  
CPU time was: 00:00:00.06
```

```
employeesalarycontrol.ctl
```

```
--options(SKIP=1, direct=y)  
--  
--c:\  
--sqlldr teaadmin/teaadmin control=C:\tea\csv\control\employeesalarycontrol.ctl log=C:\tea\csv\control\controllogs\empsalarycontrol.log  
bad=C:\tea\csv\control\controllogs\badempsalary.log  
  
options(skip=1, direct=y)  
LOAD DATA  
infile 'C:\TEA\csv\empsal.csv'  
--INFILE *  
TRUNCATE INTO TABLE teaadmin.salary  
FIELDS TERMINATED BY "," optionally enclosed by '\"'  
TRAILING NULLCOLS  
  (emp_ref integer external,  
  lastreviewdate date 'DD/MM/YYYY',  
  currmonthsal integer external,  
  prevmonthsal integer external,  
  increasereason char)
```

empsalarycontrol.log

```
SQL*Loader: Release 11.2.0.1.0 - Production on Sat Jan 7 18:52:09 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Control File: C:\tea\csv\control\employeesalarycontrol.ctl
Data File: C:\TEA\csv\empsal.csv
Bad File: C:\tea\csv\control\controllogs\badempsalary.log
Discard File: none specified

(Allow all discards)
```

```
Number to load: ALL
Number to skip: 1
Errors allowed: 50
Continuation: none specified
Path used: Direct
```

```
Table TEAADMIN.SALARY, loaded from every logical record.
Insert option in effect for this table: TRUNCATE
TRAILING NULLCOLS option in effect
```

Column Name	Position	Len	Term	Encl	Datatype
EMP_REF	FIRST	*	,	O(")	CHARACTER
LASTREVIEWDATE	NEXT	*	,	O(")	DATE DD/MM/YYYY
CURRMONTHSAL	NEXT	*	,	O(")	CHARACTER
PREVMONTHSAL	NEXT	*	,	O(")	CHARACTER
INCREASEREASON	NEXT	*	,	O(")	CHARACTER

```
Referential Integrity Constraint/Trigger Information:
NULL, UNIQUE, and PRIMARY KEY constraints are unaffected.
```

```
Constraint TEAADMIN.SALARY.EMPSAL_EMP_FK was disabled and novalidated before the load.
The following index(es) on table TEAADMIN.SALARY were processed:
index TEAADMIN.SALARY_IDX loaded successfully with 12 keys
```

```
Table TEAADMIN.SALARY has no constraint exception table.
No CHECK, REFERENTIAL constraints were re-enabled after the load.
```

```
Table TEAADMIN.SALARY:
12 Rows successfully loaded.
0 Rows not loaded due to data errors.
0 Rows not loaded because all WHEN clauses were failed.
0 Rows not loaded because all fields were null.
```

```
Date cache:
Max Size: 1000
Entries : 3
Hits   : 3
Misses : 0
```

```
Bind array size not used in direct path.  
Column array rows : 5000  
Stream buffer bytes: 256000  
Read   buffer bytes: 1048576  
  
Total logical records skipped: 1  
Total logical records read: 12  
Total logical records rejected: 0  
Total logical records discarded: 0  
Total stream buffers loaded by SQL*Loader main thread: 1  
Total stream buffers loaded by SQL*Loader load thread: 0  
  
Run began on Sat Jan 07 18:52:09 2017  
Run ended on Sat Jan 07 18:52:25 2017  
  
Elapsed time was: 00:00:15.45  
CPU time was: 00:00:00.06
```

User Creation

January 4th, 2017

Kenny Vigar

kvigar@gmail.com

Contents	Page
Script Start - Cleanup	1
Profile Creation	2
Verify Profile	6
Create User Roles	9
Verify User Roles	15
Create Users	18
Verify Users	25
Auditing Considerations	30

Cleanup all roles/profiles

```
SQL> SET VERIFY ON
SQL> TTITLE OFF
SQL> -- C:\TEA\new\Users\1_User_Create.sql
SQL>
SQL> -- quick user clean up
SQL> drop role r_manager cascade;
```

Role dropped.

```
SQL> drop role r_sragent cascade;
```

Role dropped.

```
SQL> drop role r_salesagent cascade;
```

Role dropped.

```
SQL> drop role r_commission cascade;
```

Role dropped.

```
SQL> drop role limited_READONLY cascade;
```

Role dropped.

```
SQL> drop profile jr;
```

Profile dropped.

```
SQL> drop profile salesagent;
```

Profile dropped.

```
SQL> drop profile intermediate;
```

Profile dropped.

```
SQL> drop profile mgmtandcomm;
```

Profile dropped.

```
SQL> -- TEA USER SCRIPT CREATION
SQL> --
SQL> --
SQL> -- Current User List...
SQL> SELECT username FROM DBA_USERS;
```

USERNAME
TEAADMIN
TEADBA
DBSNMP
APPQOSSYS
SYSTEM
SYS
OUTLN
ORACLE_OCM
DIP

9 rows selected.

```
SQL>
SQL>
SQL>
SQL> -----
SQL> -- CREATE USER PROFILES
SQL>
SQL> -- NOTES
SQL>      -- use to allow resource limits in profiles
SQL>
SQL>      ALTER SYSTEM SET RESOURCE_LIMIT=TRUE scope=spfile;
```

System altered.

```
SQL>
SQL>
SQL> ALTER PROFILE default LIMIT
  2  FAILED_LOGIN_ATTEMPTS 3
  3  PASSWORD_VERIFY_FUNCTION verify_function_11G
  4  PASSWORD_REUSE_MAX 2
  5  PASSWORD_REUSE_TIME 30
  6  PRIVATE_SGA UNLIMITED
  7  IDLE_TIME 30
  8  CONNECT_TIME 480
  9  LOGICAL_READS_PER_CALL 2000
10  LOGICAL_READS_PER_SESSION 2000
```

```
11 CPU_PER_CALL 2000
12 CPU_PER_SESSION 2000
13 SESSIONS_PER_USER 2
14 COMPOSITE_LIMIT 2000
15 PASSWORD_GRACE_TIME 2
16 PASSWORD_LIFE_TIME 120
17 PASSWORD_LOCK_TIME 1/24;
```

Profile altered.

```
SQL>
SQL> CREATE PROFILE jr LIMIT
  2 FAILED_LOGIN_ATTEMPTS 3
  3 PASSWORD_VERIFY_FUNCTION verify_function_11G
  4 PASSWORD_REUSE_MAX 2
  5 PASSWORD_REUSE_TIME 30
  6 PRIVATE_SGA UNLIMITED
  7 IDLE_TIME 15
  8 CONNECT_TIME 240
  9 LOGICAL_READS_PER_CALL default
10 LOGICAL_READS_PER_SESSION default
11 CPU_PER_CALL default
12 CPU_PER_SESSION default
13 SESSIONS_PER_USER 1
14 COMPOSITE_LIMIT default
15 PASSWORD_GRACE_TIME 1
16 PASSWORD_LIFE_TIME 60
17 PASSWORD_LOCK_TIME 60/1440;
```

Profile created.

```
SQL>
SQL> CREATE PROFILE salesagent LIMIT
  2 FAILED_LOGIN_ATTEMPTS 3
  3 PASSWORD_VERIFY_FUNCTION verify_function_11G
  4 PASSWORD_REUSE_MAX 2
  5 PASSWORD_REUSE_TIME 60
  6 PRIVATE_SGA UNLIMITED
  7 IDLE_TIME 45
  8 CONNECT_TIME 480
  9 LOGICAL_READS_PER_CALL 5000
10 LOGICAL_READS_PER_SESSION 5000
11 CPU_PER_CALL 5000
```

```
12 CPU_PER_SESSION 5000
13 SESSIONS_PER_USER 3
14 COMPOSITE_LIMIT 5000
15 PASSWORD_GRACE_TIME 5
16 PASSWORD_LIFE_TIME 120
17 PASSWORD_LOCK_TIME 1/24;
```

Profile created.

```
SQL>
SQL>
SQL> CREATE PROFILE intermediate LIMIT
  2 FAILED_LOGIN_ATTEMPTS 3
  3 PASSWORD_VERIFY_FUNCTION verify_function_11G
  4 PASSWORD_REUSE_MAX 2
  5 PASSWORD_REUSE_TIME 60
  6 PRIVATE_SGA UNLIMITED
  7 IDLE_TIME 45
  8 CONNECT_TIME 480
  9 LOGICAL_READS_PER_CALL 5000
10 LOGICAL_READS_PER_SESSION 5000
11 CPU_PER_CALL 5000
12 CPU_PER_SESSION 5000
13 SESSIONS_PER_USER 3
14 COMPOSITE_LIMIT 5000
15 PASSWORD_GRACE_TIME 5
16 PASSWORD_LIFE_TIME 120
17 PASSWORD_LOCK_TIME 1/24;
```

Profile created.

```
SQL>
SQL> CREATE PROFILE MGMTandComm LIMIT
  2 FAILED_LOGIN_ATTEMPTS 3
  3 PASSWORD_VERIFY_FUNCTION verify_function_11G
  4 PASSWORD_REUSE_MAX 5
  5 PASSWORD_REUSE_TIME 60
  6 PRIVATE_SGA UNLIMITED
  7 IDLE_TIME 240
  8 CONNECT_TIME 960
  9 LOGICAL_READS_PER_CALL 5000
10 LOGICAL_READS_PER_SESSION 5000
11 CPU_PER_CALL 5000
```

```
12 CPU_PER_SESSION 5000
13 SESSIONS_PER_USER 3
14 COMPOSITE_LIMIT 5000
15 PASSWORD_GRACE_TIME 7
16 PASSWORD_LIFE_TIME 240
17 PASSWORD_LOCK_TIME 2/1440;
```

Profile created.

```
SQL>
SQL>
SQL>
SQL> --ALTER PROFILE dbaprofile LIMIT
SQL> --FAILED_LOGIN_ATTEMPTS 2
SQL> --PASSWORD_VERIFY_FUNCTION verify_function_11G
SQL> --PASSWORD_REUSE_MAX 2
SQL> --PASSWORD_REUSE_TIME 90
SQL> --PRIVATE_SGA UNLIMITED
SQL> --IDLE_TIME 960
SQL> --CONNECT_TIME 960
SQL> --LOGICAL_READS_PER_CALL UNLIMITED
SQL> --LOGICAL_READS_PER_SESSION UNLIMITED
SQL> --CPU_PER_CALL UNLIMITED
SQL> --CPU_PER_SESSION UNLIMITED
SQL> --SESSIONS_PER_USER 5
SQL> --COMPOSITE_LIMIT UNLIMITED
SQL> --PASSWORD_GRACE_TIME 120
SQL> --PASSWORD_LIFE_TIME 240
SQL> --PASSWORD_LOCK_TIME 2/1440;
SQL>
SQL>
SQL>
SQL>
SQL>
```

```

SQL> -----
SQL> -- VERIFY PROFILE INFORMATION
SQL> -----
SQL>
SQL> SET LINESIZE 80
SQL> SET PAGESIZE 130
SQL> TTITLE 'DBA Profile Review'
SQL> BREAK ON PROFILE ON resource skip 1
SQL> COL "PROFILE" FORMAT a13
SQL> COL "RESOURCE" FORMAT a11
SQL> COL "LIMIT VALUE" FORMAT a19
SQL> SELECT profile,resource_type "RESOURCE", resource_name ,limit "limit value"
  2  FROM DBA_PROFILES
  3  ORDER BY profile asc, resource_type;

```

Sun Jan 08

page 1

DBA Profile Review

PROFILE	RESOURCE	RESOURCE_NAME	limit value
DBaprofile	KERNEL	CPU_PER_CALL	UNLIMITED
		PRIVATE_SGA	UNLIMITED
		SESSIONS_PER_USER	5
		IDLE_TIME	960
		CPU_PER_SESSION	UNLIMITED
		LOGICAL_READS_PER_CALL	UNLIMITED
		LOGICAL_READS_PER_SESSION	UNLIMITED
		COMPOSITE_LIMIT	UNLIMITED
		CONNECT_TIME	960
		PASSWORD_REUSE_MAX	2
PASSWORD	PASSWORD	FAILED_LOGIN_ATTEMPTS	2
		PASSWORD_VERIFY_FUNCTION	VERIFY_FUNCTION_11G
		PASSWORD_REUSE_TIME	90
		PASSWORD_GRACE_TIME	120
		PASSWORD_LIFE_TIME	240
		PASSWORD_LOCK_TIME	.0013
DEFAULT	KERNEL	COMPOSITE_LIMIT	2000
		PRIVATE_SGA	UNLIMITED
		SESSIONS_PER_USER	2
		IDLE_TIME	30
		CPU_PER_CALL	2000
		CPU_PER_SESSION	2000
		LOGICAL_READS_PER_CALL	2000
		LOGICAL_READS_PER_SESSION	2000
		CONNECT_TIME	480

	PASSWORD	PASSWORD_VERIFY_FUNCTION FAILED_LOGIN_ATTEMPTS PASSWORD_REUSE_TIME PASSWORD_LOCK_TIME PASSWORD_LIFE_TIME PASSWORD_GRACE_TIME PASSWORD_REUSE_MAX	VERIFY_FUNCTION_11G 3 30 .0416 120 2 2
INTERMEDIATE	KERNEL	LOGICAL_READS_PER_CALL IDLE_TIME PRIVATE_SGA SESSIONS_PER_USER COMPOSITE_LIMIT CONNECT_TIME CPU_PER_CALL CPU_PER_SESSION LOGICAL_READS_PER_SESSION	5000 45 UNLIMITED 3 5000 480 5000 5000 5000
	PASSWORD	PASSWORD_VERIFY_FUNCTION FAILED_LOGIN_ATTEMPTS PASSWORD_GRACE_TIME PASSWORD_LOCK_TIME PASSWORD_LIFE_TIME PASSWORD_REUSE_MAX PASSWORD_REUSE_TIME	VERIFY_FUNCTION_11G 3 5 .0416 120 2 60
JR	KERNEL	CONNECT_TIME SESSIONS_PER_USER IDLE_TIME PRIVATE_SGA COMPOSITE_LIMIT CPU_PER_SESSION CPU_PER_CALL LOGICAL_READS_PER_SESSION LOGICAL_READS_PER_CALL	240 1 15 UNLIMITED DEFAULT DEFAULT DEFAULT DEFAULT DEFAULT
	PASSWORD	PASSWORD_GRACE_TIME PASSWORD_LOCK_TIME PASSWORD_REUSE_TIME FAILED_LOGIN_ATTEMPTS PASSWORD_VERIFY_FUNCTION PASSWORD_LIFE_TIME PASSWORD_REUSE_MAX	1 .0416 30 3 VERIFY_FUNCTION_11G 60 2
MGMTANDCOMM	KERNEL	COMPOSITE_LIMIT LOGICAL_READS_PER_CALL CONNECT_TIME	5000 5000 960

		IDLE_TIME	240
		LOGICAL_READS_PER_SESSION	5000
		PRIVATE_SGA	UNLIMITED
		CPU_PER_SESSION	5000
		CPU_PER_CALL	5000
		SESSIONS_PER_USER	3
	PASSWORD	PASSWORD_VERIFY_FUNCTION	VERIFY_FUNCTION_11G
		PASSWORD_REUSE_MAX	5
		PASSWORD_GRACE_TIME	7
		PASSWORD_LOCK_TIME	.0013
		PASSWORD_REUSE_TIME	60
		PASSWORD_LIFE_TIME	240
		FAILED_LOGIN_ATTEMPTS	3
SALESAGENT	KERNEL	LOGICAL_READS_PER_CALL	5000
		LOGICAL_READS_PER_SESSION	5000
		CPU_PER_SESSION	5000
		COMPOSITE_LIMIT	5000
		CONNECT_TIME	480
		SESSIONS_PER_USER	3
		IDLE_TIME	45
		CPU_PER_CALL	5000
		PRIVATE_SGA	UNLIMITED
	PASSWORD	PASSWORD_LIFE_TIME	120
		PASSWORD_VERIFY_FUNCTION	VERIFY_FUNCTION_11G
		PASSWORD_GRACE_TIME	5
		FAILED_LOGIN_ATTEMPTS	3
		PASSWORD_LOCK_TIME	.0416
		PASSWORD_REUSE_MAX	2
		PASSWORD_REUSE_TIME	60

96 rows selected.

SQL> Prompt ****

SQL> PROMPT

SQL> PROMPT Profiles complete
Profiles complete
SQL> PROMPT

SQL> Prompt ****

SQL> PAUSE

```
SQL> -----
SQL> --
SQL> -- Create User Roles
SQL> --
SQL> --
SQL>
SQL> --
SQL> -- Limited - read only for JR Users
SQL> --      - Select on teaadmin..customer, and teaadmin.customerdetail,
SQL> --      - Insert on teaadmin.customercomm for marketing assignments.
SQL>
SQL> CREATE ROLE limited_READONLY;

Role created.

SQL> GRANT create session to limited_READONLY;

Grant succeeded.

SQL> grant select on teaadmin.customer to limited_READONLY;

Grant succeeded.

SQL> grant select on teaadmin.customerdetail to limited_READONLY;

Grant succeeded.

SQL> grant select on teaadmin.customercomm to limited_READONLY;

Grant succeeded.

SQL>
SQL> -- Sales Agent - Select, Insert, Update on Customer information for editing records)
SQL> --          - Select, Insert, Update on Itinerary, Payment, and Sale
SQL> --
SQL> --          - Select on Supplier and Product data
SQL> --          - Select on reference tables, Fee, Destination, Class, Affiliation
SQL>
SQL> CREATE ROLE r_salesagent;

Role created.

SQL> GRANT CONNECT to r_salesagent;
```

```
Grant succeeded.

SQL> GRANT SELECT, INSERT, UPDATE ON teaadmin.customer TO r_salesagent;
Grant succeeded.

SQL> GRANT SELECT, INSERT, UPDATE ON teaadmin.customerdetail TO r_salesagent;
Grant succeeded.

SQL> GRANT SELECT, INSERT, UPDATE ON teaadmin.customercomm TO r_salesagent;
Grant succeeded.

SQL>
SQL> GRANT SELECT, UPDATE, INSERT ON teaadmin.itinerary TO r_salesagent;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT ON teaadmin.payment TO r_salesagent;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT ON teaadmin.sale TO r_salesagent;
Grant succeeded.

SQL>
SQL> GRANT SELECT ON teaadmin.supplier TO r_salesagent;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.supplierdetail TO r_salesagent;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.affiliation TO r_salesagent;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.product TO r_salesagent;
```

```
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.destination TO r_salesagent;

Grant succeeded.

SQL> GRANT SELECT ON teaadmin.fee TO r_salesagent;

Grant succeeded.

SQL> GRANT SELECT ON teaadmin.travelclass TO r_salesagent;

Grant succeeded.

SQL>
SQL>
SQL> -- Sr Agent      - Same privileges as r_salesagent, but with Delete for Itinerary, and Customer Information.
SQL> --                  - Not able to delete Payments, or Sale.
SQL> --
SQL>
SQL> CREATE ROLE r_sragent;

Role created.

SQL> GRANT CONNECT to r_sragent;

Grant succeeded.

SQL> GRANT r_salesagent to r_sragent;

Grant succeeded.

SQL>
SQL> GRANT DELETE ON teaadmin.customer TO r_sragent;

Grant succeeded.

SQL> GRANT DELETE ON teaadmin.customerdetail TO r_sragent;

Grant succeeded.

SQL> GRANT DELETE ON teaadmin.customercomm TO r_sragent;
```

```
Grant succeeded.

SQL> GRANT DELETE ON teaadmin.itinerary TO r_sragent;

Grant succeeded.

SQL>
SQL> -- Commission Specialist - Full permissions to Commissions table, in addition to r_sragent
SQL> --
SQL> CREATE ROLE r_commission;

Role created.

SQL> GRANT CONNECT to r_commission;

Grant succeeded.

SQL> GRANT r_sragent to r_commission;

Grant succeeded.

SQL>
SQL> GRANT ALL ON teaadmin.commission TO r_commission;

Grant succeeded.

SQL>
SQL>
SQL> -- Manager - Full Table Permissions as requested by TEA.
SQL> --           - Do not want to implicitly grant 'all'
SQL>
SQL>
SQL> CREATE ROLE r_manager
  2 identified by teamanager;

Role created.

SQL>
SQL> GRANT CONNECT to r_manager;

Grant succeeded.
```

```
SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.affiliation TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.cc TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.commission TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.customer TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.customerdetail TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.customercomm TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.destination TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.employee TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.employeedetail TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.salary TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.fee TO r_manager;
Grant succeeded.
```

```
SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.itinerary TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.payment TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.product TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.sale TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.supplier TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.supplierdetail TO r_manager;
Grant succeeded.

SQL> GRANT SELECT, UPDATE, INSERT, DELETE ON teaadmin.travelclass TO r_manager;
Grant succeeded.
```

```

SQL>
SQL>
SQL> TTITLE "Role Prvs Report"
SQL> set linesize 70
SQL> set pagesize 40
SQL> COL grantee format a17
SQL> COL "Table Prvs" format a40
SQL> BREAK ON "grantee" skip 1
SQL> SELECT grantee, ' has || privilege ||' on ' ||table_name "Table Prvs"
  2  FROM dba_tab_privs where owner NOT IN ('SYS', 'DBSNMP', 'APPQOSSYS','SYSTEM','OUTLN')
  3  ORDER BY grantee;

```

Sun Jan 08 page 1
 Role Prvs Report

GRANTEE	Table Prvs
LIMITED_READONLY	has SELECT on CUSTOMERCOMM has SELECT on CUSTOMER has SELECT on CUSTOMERDETAIL
R_COMMISION	has ON COMMIT REFRESH on COMMISSION has UPDATE on COMMISSION has INSERT on COMMISSION has DELETE on COMMISSION has ALTER on COMMISSION has QUERY REWRITE on COMMISSION has DEBUG on COMMISSION has FLASHBACK on COMMISSION has SELECT on COMMISSION
R_MANAGER	has UPDATE on AFFILIATION has SELECT on AFFILIATION has INSERT on AFFILIATION has DELETE on AFFILIATION has UPDATE on CC has SELECT on CC has INSERT on CC has DELETE on CC has UPDATE on COMMISSION has SELECT on COMMISSION has INSERT on COMMISSION has DELETE on COMMISSION has UPDATE on CUSTOMER has SELECT on CUSTOMER has INSERT on CUSTOMER has DELETE on CUSTOMER

```
has UPDATE on CUSTOMERCOMM
has SELECT on CUSTOMERCOMM
has INSERT on CUSTOMERCOMM
has DELETE on CUSTOMERCOMM

R_MANAGER
has UPDATE on CUSTOMERDETAIL
has SELECT on CUSTOMERDETAIL
has INSERT on CUSTOMERDETAIL
has DELETE on CUSTOMERDETAIL
has UPDATE on DESTINATION
has SELECT on DESTINATION
has INSERT on DESTINATION
has DELETE on DESTINATION
has UPDATE on EMPLOYEE
has SELECT on EMPLOYEE
has INSERT on EMPLOYEE
has DELETE on EMPLOYEE
has UPDATE on EMPLOYEEDETAIL
has SELECT on EMPLOYEEDETAIL
has INSERT on EMPLOYEEDETAIL
has DELETE on EMPLOYEEDETAIL
has UPDATE on FEE
has SELECT on FEE
has INSERT on FEE
has DELETE on FEE
has UPDATE on ITINERARY
has SELECT on ITINERARY
has INSERT on ITINERARY
has DELETE on ITINERARY
has UPDATE on PAYMENT
has SELECT on PAYMENT
has INSERT on PAYMENT
has DELETE on PAYMENT
has UPDATE on PRODUCT
has SELECT on PRODUCT
has INSERT on PRODUCT
has DELETE on PRODUCT
has UPDATE on SALARY
has SELECT on SALARY

R_MANAGER
has INSERT on SALARY
has DELETE on SALARY
has UPDATE on SALE
has SELECT on SALE
has INSERT on SALE
has DELETE on SALE
has UPDATE on SUPPLIER
has SELECT on SUPPLIER
```

```

has INSERT on SUPPLIER
has DELETE on SUPPLIER
has UPDATE on SUPPLIERDETAIL
has SELECT on SUPPLIERDETAIL
has INSERT on SUPPLIERDETAIL
has DELETE on SUPPLIERDETAIL
has UPDATE on TRAVELCLASS
has SELECT on TRAVELCLASS
has INSERT on TRAVELCLASS
has DELETE on TRAVELCLASS

R_SALESAGENT has SELECT on AFFILIATION
R_SALESAGENT has SELECT on FEE
R_SALESAGENT has SELECT on SUPPLIERDETAIL
R_SALESAGENT has SELECT on SUPPLIER
R_SALESAGENT has INSERT on SALE
R_SALESAGENT has UPDATE on CUSTOMER
R_SALESAGENT has SELECT on CUSTOMER
R_SALESAGENT has INSERT on CUSTOMER
R_SALESAGENT has UPDATE on CUSTOMERCOMM
R_SALESAGENT has SELECT on CUSTOMERCOMM
R_SALESAGENT has INSERT on CUSTOMERCOMM
R_SALESAGENT has UPDATE on CUSTOMERDETAIL
R_SALESAGENT has SELECT on CUSTOMERDETAIL
R_SALESAGENT has INSERT on CUSTOMERDETAIL
R_SALESAGENT has SELECT on DESTINATION

R_SALESAGENT has SELECT on TRAVELCLASS
R_SALESAGENT has UPDATE on ITINERARY
R_SALESAGENT has SELECT on ITINERARY
R_SALESAGENT has INSERT on ITINERARY
R_SALESAGENT has UPDATE on PAYMENT
R_SALESAGENT has SELECT on PAYMENT
R_SALESAGENT has INSERT on PAYMENT
R_SALESAGENT has SELECT on PRODUCT
R_SALESAGENT has UPDATE on SALE
R_SALESAGENT has SELECT on SALE

R_SRAGENT has DELETE on CUSTOMERDETAIL
R_SRAGENT has DELETE on ITINERARY
R_SRAGENT has DELETE on CUSTOMERCOMM
R_SRAGENT has DELETE on CUSTOMER

```

113 rows selected.

SQL> clear break;
SQL> ttitle off

User Creation

```
SQL> SET ECHO ON
SQL> SET VERIFY ON
SQL> TTITLE OFF
SQL>
SQL> -- C:\TEA\new\Users\2_User_Create.sql
SQL>
SQL> -----
SQL> --
SQL> -- User Creation continued
SQL> --
SQL> -- Creating users currently on TEA Employee List.

SQL> -----
SQL> --TABLE OWNER: TEAADMIN
SQL> -- For reference, commenting out user Tea Admin. Will not drop and reassign because we build all the
-- tables with this
SQL> -- table owner.

SQL>
SQL>
SQL> --create user teaadmin
SQL> --identified by teadadmin
SQL> --default tablespace userdata
SQL> --temporary tablespace temp
SQL> --quota 30m on userdata
SQL> --quota 30m on supplierdata
SQL> --quota 10m on indx
SQL> --quota 10m on adhoc
SQL> --quota 10m on readonly;
SQL> --grant connect to teaadmin;
SQL> --grant create table to teaadmin;
SQL> --grant drop any table to teaadmin;
SQL> --GRANT create view to teaadmin;
SQL>
SQL> --GRANT all ON tempdata to TEAADMIN;
SQL> --GRANT all ON tempsuppdata to TEAADMIN;
SQL> --GRANT all ON tempsalesdata to TEAADMIN;
SQL> --GRANT EXP_FULL_DATABASE
SQL> --GRANT IMP_FULL_DATABASE
SQL>
SQL>
SQL> -----
```

```
SQL> -- TEA USERS
SQL>
SQL> --Jr Agents:
SQL>
SQL> -- Janet Delton
SQL>
SQL> create user jdelton
  2  identified by jdelton2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile jr;
```

User created.

```
SQL>
SQL> grant limited_READONLY to jdelton;
```

Grant succeeded.

```
SQL>
SQL>
SQL> -- Judy Lisle
SQL>
SQL> create user jlisle
  2  identified by jlisle2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile jr;
```

User created.

```
SQL>
SQL> grant limited_READONLY to jdelton;
```

Grant succeeded.

```
SQL>
SQL> -- Dennis Reynolds
SQL>
SQL> create user dreynolds
  2  identified by dreynolds2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile jr;
```

User created.

```
SQL>
SQL> grant limited_READONLY to jdelton;
```

Grant succeeded.

```
SQL> -- Sales Agents
SQL>
SQL> -- John Coville
SQL>
SQL> create user jcoville
  2  identified by jcoville2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile salesagent;
```

User created.

```
SQL>
SQL> GRANT r_salesagent TO jcoville;
```

Grant succeeded.

```
SQL>
SQL> -- Janice Dhal
SQL>
SQL> create user jdhal
  2  identified by jdhal2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile salesagent;
```

User created.

```
SQL>
SQL> GRANT r_salesagent TO jdhal;
```

Grant succeeded.

```
SQL>
SQL> -- Bruce Dixon
SQL>
SQL> create user bdixon
  2  identified by bdixon2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile salesagent;
```

User created.

```
SQL>
SQL> GRANT r_salesagent TO bdixon;
```

Grant succeeded.

```
SQL> -- Beverly Jones
SQL>
SQL> create user bjones
  2  identified by bjones2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile salesagent;
```

User created.

```
SQL>
SQL> GRANT r_salesagent TO bjones;
```

Grant succeeded.

```
SQL>
SQL>
SQL> -- Intermediate Sales Agents
SQL>
SQL> --Brian Peterson
SQL>
SQL> create user bpeterson
  2  identified by bpeterson2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile intermediate;
```

User created.

```
SQL>
SQL> GRANT r_salesagent TO bpeterson;
```

Grant succeeded.

```
SQL>
SQL> -- Jane Merill
SQL>
SQL> create user jmerrill
  2  identified by jmerrill2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile intermediate;
```

User created.

```
SQL>
SQL> GRANT r_salesagent TO jmerrill;
```

Grant succeeded.

```
SQL>
SQL> -- OWNER
SQL>
SQL> -- Karen Graham
SQL> create user kgraham
  2  identified by kgraham2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile mgmtandcomm;
```

User created.

```
SQL>
SQL> -- Request from TEA to allow only Select to owner
SQL>
SQL> GRANT CONNECT to kgraham;
```

Grant succeeded.

```
SQL>
SQL>
SQL> -- Commission Specialist
SQL>
SQL> create user jolandaw
  2  identified by jolandaw2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile mgmtandcomm;
```

User created.

```
SQL>
SQL> GRANT r_commission TO jolandaw;
```

Grant succeeded.

```
SQL>
SQL>
SQL> -- Manager
SQL>
SQL>
SQL> create user gsmith
  2  identified by seahawks2016
  3  default tablespace userdata
  4  temporary tablespace temp
  5  account lock
  6  password expire
  7  profile mgmtandcomm;
```

User created.

```
SQL>
SQL> GRANT r_manager TO gsmith;
```

Grant succeeded.

```
SQL>
SQL>
SQL>
```

Verification

```

SQL> SET ECHO ON
SQL> SET VERIFY ON
SQL> TTITLE OFF
SQL>
SQL> -- C:\TEA\new\Users\3_User_Create.sql
SQL> -- DBA USERS
SQL> -----
SQL> CLEAR BREAK
SQL> SET linesize 100
SQL> TTITLE CENTER '** Current DBA Users **'
SQL> REPHEADER CENTER '(not including users SYS, DBSNMP, APPQOSSYS, SYSTEM, OUTLN, DIP, ORACLE_OCM)'
SQL> COL username FORMAT a15
SQL> col account_status FORMAT a20
SQL> col default_tablespace FORMAT a15
SQL> col temporary_tablespace FORMAT a10
SQL> col profile FORMAT a15
SQL> SELECT username, profile, account_status, default_tablespace, temporary_tablespace, created
  2  FROM dba_users WHERE username NOT IN ('SYS', 'DBSNMP', 'APPQOSSYS', 'SYSTEM', 'OUTLN', 'DIP', 'ORACLE_OCM')
  3  ORDER BY created asc ;

          ** Current DBA Users **
          (not including users SYS, DBSNMP, APPQOSSYS, SYSTEM, OUTLN, DIP, ORACLE_OCM)
-----  

USERNAME      PROFILE     ACCOUNT_STATUS    DEFAULT_TABLESP TEMPORARY_ CREATED
-----  

TEADBA        DBAPROFILE  OPEN             USERDATA        TEMP       28-NOV-16
TEAADMIN      DBAPROFILE  OPEN             USERDATA        TEMP       02-DEC-16
JDELTON       JR           EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
JMERRILL      INTERMEDIATE EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
DREYNOLDS     JR           EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
BDIXON        SALESAGENT   EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
BJONES         SALESAGENT   EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
JDHAL         SALESAGENT   EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
JCOVILLE     SALESAGENT   EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
JLISLE         JR           EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
JOLANDAW      MGMTANDCOMM EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
KGRAHAM        MGMTANDCOMM EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
GSMITH         MGMTANDCOMM EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17
BPETERSON     INTERMEDIATE EXPIRED & LOCKED  USERDATA        TEMP       08-JAN-17

14 rows selected.
SQL> REPHEADER OFF
SQL>

```

```
SQL>
SQL>
SQL> -----
SQL> -- ROLE PRIVILEGES
SQL> -----
SQL>
SQL>
SQL> TTITLE "Role Privils Report"
SQL> set linesize 70
SQL> set pagesize 150
SQL> COL grantee format a17
SQL> COL "Table Privils" format a40
SQL> BREAK ON "grantee" skip 1
SQL> SELECT grantee, ' has '''|| privilege ||' on ' ||table_name "Table Privils"
  2  FROM dba_tab_privs where owner NOT IN ('SYS', 'DBSNMP', 'APPQOSSYS','SYSTEM','OUTLN')
  3 ORDER BY grantee;
```

Sun Jan 08 page 1
Role Privs Report

GRANTEE	Table Privs
LIMITED_READONLY	has SELECT on CUSTOMERCOMM has SELECT on CUSTOMER has SELECT on CUSTOMERDETAIL
R_COMMISSION	has ON COMMIT REFRESH on COMMISSION has UPDATE on COMMISSION has INSERT on COMMISSION has DELETE on COMMISSION has ALTER on COMMISSION has QUERY REWRITE on COMMISSION has DEBUG on COMMISSION has FLASHBACK on COMMISSION has SELECT on COMMISSION
R_MANAGER	has UPDATE on AFFILIATION has SELECT on AFFILIATION has INSERT on AFFILIATION has DELETE on AFFILIATION has UPDATE on CC has SELECT on CC has INSERT on CC has DELETE on CC has UPDATE on COMMISSION has SELECT on COMMISSION

```
has INSERT on COMMISSION
has DELETE on COMMISSION
has UPDATE on CUSTOMER
has SELECT on CUSTOMER
has INSERT on CUSTOMER
has DELETE on CUSTOMER
has UPDATE on CUSTOMERCOMM
has SELECT on CUSTOMERCOMM
has INSERT on CUSTOMERCOMM
has DELETE on CUSTOMERCOMM
has UPDATE on CUSTOMERDETAIL
has SELECT on CUSTOMERDETAIL
has INSERT on CUSTOMERDETAIL
has DELETE on CUSTOMERDETAIL
has UPDATE on DESTINATION
has SELECT on DESTINATION
has INSERT on DESTINATION
has DELETE on DESTINATION
has UPDATE on EMPLOYEE
has SELECT on EMPLOYEE
has INSERT on EMPLOYEE
has DELETE on EMPLOYEE
has UPDATE on EMPLOYEEDETAIL
has SELECT on EMPLOYEEDETAIL
has INSERT on EMPLOYEEDETAIL
has DELETE on EMPLOYEEDETAIL
has UPDATE on FEE
has SELECT on FEE
has INSERT on FEE
has DELETE on FEE
has UPDATE on ITINERARY
has SELECT on ITINERARY
has INSERT on ITINERARY
has DELETE on ITINERARY
has UPDATE on PAYMENT
has SELECT on PAYMENT
has INSERT on PAYMENT
has DELETE on PAYMENT
has UPDATE on PRODUCT
has SELECT on PRODUCT
has INSERT on PRODUCT
has DELETE on PRODUCT
has UPDATE on SALARY
has SELECT on SALARY
has INSERT on SALARY
has DELETE on SALARY
has UPDATE on SALE
has SELECT on SALE
```

```

has INSERT on SALE
has DELETE on SALE
has UPDATE on SUPPLIER
has SELECT on SUPPLIER
has INSERT on SUPPLIER
has DELETE on SUPPLIER
has UPDATE on SUPPLIERDETAIL
has SELECT on SUPPLIERDETAIL
has INSERT on SUPPLIERDETAIL
has DELETE on SUPPLIERDETAIL
has UPDATE on TRAVELCLASS
has SELECT on TRAVELCLASS
has INSERT on TRAVELCLASS
has DELETE on TRAVELCLASS

R_SALESAGENT      has SELECT on AFFILIATION
                  has SELECT on FEE
                  has SELECT on SUPPLIERDETAIL
                  has SELECT on SUPPLIER
                  has INSERT on SALE
                  has UPDATE on CUSTOMER
                  has SELECT on CUSTOMER
                  has INSERT on CUSTOMER
                  has UPDATE on CUSTOMERCOMM
                  has SELECT on CUSTOMERCOMM
                  has INSERT on CUSTOMERCOMM
                  has UPDATE on CUSTOMERDETAIL
                  has SELECT on CUSTOMERDETAIL
                  has INSERT on CUSTOMERDETAIL
                  has SELECT on DESTINATION
                  has SELECT on TRAVELCLASS
                  has UPDATE on ITINERARY
                  has SELECT on ITINERARY
                  has INSERT on ITINERARY
                  has UPDATE on PAYMENT
                  has SELECT on PAYMENT
                  has INSERT on PAYMENT
                  has SELECT on PRODUCT
                  has UPDATE on SALE
                  has SELECT on SALE

R_SRAGENT         has DELETE on CUSTOMERDETAIL
                  has DELETE on ITINERARY
                  has DELETE on CUSTOMERCOMM
                  has DELETE on CUSTOMER

```

113 rows selected.

```
SQL> clear break;
SQL>
SQL> -----
SQL> -- DBA ROLE PRIVS (Admin user verification)
SQL>
SQL> BREAK ON grantee SKIP 1
SQL> SELECT grantee, granted_role, admin_option
  2  FROM dba_role_privs
  3  where grantee IN ('TEAADMIN', 'TEADBA', 'TJACKSON', 'RWILSON', 'SENIOR')
  4  order by grantee;
```

Sun Jan 08 page 1
Role Privils Report

GRANTEE	GRANTED_ROLE	ADM
TEAADMIN	EXP_FULL_DATABASE	NO
	IMP_FULL_DATABASE	NO
TEADBA	DBA	NO

```
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

Auditing Considerations

Forward

Auditing allows us to track any failed, or successful activity on tables within the TEA database. This document highlights the steps to enable auditing, and some suggestions on what should be audited. It should also be noted, that any auditing on the database will cause an increased load on the system, and should be used only on critical tables.

1. Steps to Enable Audit

Recipe

-- Login as SYS/oracle as SYSDBA

Run each command below

```
alter system set audit_trail=db scope=spfile;
```

```
alter system set audit_sys_operations=true scope=spfile;
```

(if you want to enable all sys operations audit)

```
show parameter AUDIT_TRAIL=os or =db or =false (static, will need to bounce instance)
```

-- Shut down database

-- Alter PFILE with line(s)

```
AUDIT_SYS_OPERATIONS=TRUE
```

(if you want to monitor all levels of auditing on sys users)

-- Start the database with new PFILE change

2. Code to Create Audits

By table

```
AUDIT permission  
ON table  
BY ACCESS  
WHENEVER SUCCESSFUL;
```

By users

```
AUDIT select table  
BY username  
BY ACCESS  
WHENEVER SUCCESSFUL;
```

3. Auditing Recommendations

First and foremost, we should setup auditing on our audit trail (sys.aud\$)

```
AUDIT delete, insert, update, select ON sys.aud$ BY ACCESS;
```

As well, other tables we should audit, would be any employee accessing the following critical information tables,

EMPLOYEE

SALARY

CREDITCARD (CC)

Anyone trying to access the Supplier/Products tables (which only have select provided to Agent level employees).

SUPPLIER

SUPPLIERDETAIL

PRODUCTS

Column level tracking can be provided with an FGA Policy, but at this time, there doesn't seem to be a need for that level of detail. We will confirm with TEA going forward if there are changes to this policy.

For any Audit reporting, we can use the following tables for report building and audit verification.

dba_audit_trail

sys.aud\$

dba_audit_exists

DBA_FGA_AUDIT_TRAIL

DBA_COMMON_AUDIT_TRAIL

Network Strategy

January 26th, 2017

Kenny Vigar

kvigar@gmail.com

The planning strategy for accessing the Travel Experts Travel Agency (TEA) database is listed below. Further details on each phase of the plan will be covered in its relative heading later in this document.

Contents	Page
Prepare User Account for Testing Network	1
Verify User	8
Steps for Server Side Network Configuration	12
Verify Server Settings	13
Test Results (command line copy)	16
TNSNAMES.ORA Configuration	19
LISTENER.ORA Configuration	21
SQLNET.ORA Configuration	22

```

SQL> -- User Creation for TEA Network Testing
SQL> -- Jan 2016
SQL> -- Last Updated by Kenny Vigar
SQL> --
SQL> -- This script will create a test user, profile and permissions
SQL> -- for testing remote access to the TEA Database.
SQL> --
SQL> -- User for test inserts is "Agent"
SQL> --
SQL> SET ECHO ON
SQL> SET VERIFY ON
SQL> TTITLE OFF
SQL>
SQL>
SQL> conn / as sysdba;
Connected.
SQL>
SQL>
SQL> SELECT '1 Database Name: '||name FROM v$database
  2 UNION
  3 SELECT '2 Instance Name: '|| instance_name FROM v$instance
  4 UNION
  5 SELECT '3 Host Name: '|| host_name FROM v$instance
  6 UNION
  7 SELECT '4 Todays date : '||sysdate FROM dual;

'1DATABASENAME:'||NAME
-----
1 Database Name: TEA
2 Instance Name: tea
3 Host Name: ICTVM-B6GE6VEST
4 Todays date : 26-JAN-17

SQL>
SQL> ttitle off
SQL>
SQL> -- Cleanup
SQL>
SQL> drop profile p_salesagent1 cascade;

Profile dropped.

SQL> drop role r_salesagent1 cascade;

```

Role dropped.

```
SQL> drop user Agent cascade;
```

User dropped.

```
SQL> drop table teaadmin.kennynetworktest cascade constraints;
```

Table dropped.

```
SQL>
```

```
SQL> CREATE TABLE teaadmin.kennynetworktest
  2  (ref number(3) CONSTRAINT tempref_pk PRIMARY KEY deferrable initially immediate
  3    USING INDEX (CREATE INDEX teaadmin.tempref_idx
  4                  ON teaadmin.kennynetworktest(ref)
  5                  PCTFREE 10 INITTRANS 2
  6                  STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1
  7                  MAXEXTENTS 40 PCTINCREASE 5 )
  8                  TABLESPACE INDX),
  9  first_name varchar2(50))
10  TABLESPACE COMPANYDATA
11  PCTFREE 20
12  PCTUSED 60
13  INITTRANS 2
14  STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);
```

Table created.

```
SQL> COMMENT ON TABLE teaadmin.employee IS 'Table for testing remote access inserts';
```

Comment created.

```
SQL>
SQL>
SQL> -- Create Profile for User
SQL>
SQL> CREATE PROFILE p_salesagent1 LIMIT
  2  FAILED_LOGIN_ATTEMPTS 3
  3  PASSWORD_VERIFY_FUNCTION verify_function_11G
  4  PASSWORD_REUSE_MAX 2
  5  PASSWORD_REUSE_TIME 60
  6  PRIVATE_SGA 500
  7  IDLE_TIME 45
  8  CONNECT_TIME 480
  9  LOGICAL_READS_PER_CALL 5000
10  LOGICAL_READS_PER_SESSION 5000
11  CPU_PER_CALL 5000
12  CPU_PER_SESSION 5000
13  SESSIONS_PER_USER 3
14  COMPOSITE_LIMIT 5000
15  PASSWORD_GRACE_TIME 5
16  PASSWORD_LIFE_TIME 120
17  PASSWORD_LOCK_TIME 1/24;
```

Profile created.

```
SQL>
SQL> -- Verify Profile creation
SQL>
SQL> SET LINESIZE 80
SQL> SET PAGESIZE 130
SQL> TTITLE 'Profile Review'
SQL> BREAK ON PROFILE ON resource skip 1
SQL> COL "PROFILE" FORMAT a13
SQL> COL "RESOURCE" FORMAT a11
SQL> COL "LIMIT VALUE" FORMAT a19
SQL> SELECT profile,resource_type "RESOURCE", resource_name ,limit "limit value"
  2  FROM DBA_PROFILES WHERE profile='P_SALESAGENT1'
  3  ORDER BY profile asc, resource_type;
```

Thu Jan 26

page 1

Profile Review

PROFILE	RESOURCE	RESOURCE_NAME	limit	value
P_SALESAGENT1	KERNEL	IDLE_TIME	45	
		PRIVATE_SGA	500	
		COMPOSITE_LIMIT	5000	
		CPU_PER_SESSION	5000	
		CPU_PER_CALL	5000	
		LOGICAL_READS_PER_SESSION	5000	
		LOGICAL_READS_PER_CALL	5000	
		SESSIONS_PER_USER	3	
		CONNECT_TIME	480	
	PASSWORD	PASSWORD_VERIFY_FUNCTION	VERIFY_FUNCTION_11G	
		PASSWORD_REUSE_TIME	60	
		PASSWORD_REUSE_MAX	2	
		PASSWORD_GRACE_TIME	5	
		FAILED_LOGIN_ATTEMPTS	3	
		PASSWORD_LIFE_TIME	120	
		PASSWORD_LOCK_TIME	.0416	

16 rows selected.

```
SQL>
SQL> -- Create ROLE
SQL>
SQL> CREATE ROLE r_salesagent1;
```

Role created.

```
SQL>
SQL> GRANT CONNECT to r_salesagent1;
```

Grant succeeded.

```
SQL> GRANT SELECT ON teaadmin.AFFILIATION TO r_salesagent1;
```

Grant succeeded.

```
SQL> GRANT SELECT ON teaadmin.CC TO r_salesagent1;
```

Grant succeeded.

```
SQL> GRANT SELECT ON teaadmin.COMMISSION TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.CUSTOMER TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.CUSTOMERCOMM TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.CUSTOMERDETAIL TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.DESTINATION TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.EMPLOYEE TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.EMPLOYEEDETAIL TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.FEE TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.ITINERARY TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.PAYMENT TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.PRODUCT TO r_salesagent1;
Grant succeeded.
```

```
SQL> GRANT SELECT ON teaadmin.SALARY TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.SALE TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.SUPPLIER TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.SUPPLIERDETAIL TO r_salesagent1;
Grant succeeded.

SQL> GRANT SELECT ON teaadmin.TRAVELCLASS TO r_salesagent1;
Grant succeeded.

SQL>
SQL> GRANT select, insert, update ON teaadmin.kennynetworktest TO r_salesagent1;
Grant succeeded.
```

```

SQL>
SQL> -- Verify Privs
SQL>
SQL> TTITLE "Role Privs Report"
SQL> set linesize 70
SQL> set pagesize 40
SQL> COL grantee format a17
SQL> COL "Table Privs" format a40
SQL> BREAK ON "grantee" skip 1
SQL>
SQL> SELECT grantee, ' has ''' privilege ||' on ' ||table_name "Table Privs"
2  FROM dba_tab_privs where GRANTEE = 'R_SALESAGENT1';

```

Thu Jan 26 page 1
 Role Privs Report

GRANTEE	Table Privs
R_SALESAGENT1	has SELECT on AFFILIATION has SELECT on CC has SELECT on COMMISSION has SELECT on CUSTOMER has SELECT on CUSTOMERCOMM has SELECT on CUSTOMERDETAIL has SELECT on DESTINATION has SELECT on EMPLOYEE has SELECT on EMPLOYEEDETAIL has SELECT on FEE has SELECT on ITINERARY has INSERT on KENNYSNETWORKTEST has SELECT on KENNYSNETWORKTEST has UPDATE on KENNYSNETWORKTEST has SELECT on PAYMENT has SELECT on PRODUCT has SELECT on SALARY has SELECT on SALE has SELECT on SUPPLIER has SELECT on SUPPLIERDETAIL has SELECT on TRAVELCLASS

21 rows selected.

```

SQL>
SQL> clear break;
SQL> ttitle off

```

```

SQL>
SQL> -- Create User
SQL>
SQL> create user Agent
  2  identified by Agent28#
  3  default tablespace userdata
  4  temporary tablespace temp
  5  quota 10m on adhoc
  6  profile p_salesagent1;

User created.

SQL>
SQL> grant r_salesagent1 to agent;

Grant succeeded.

SQL>
SQL> -- Verify User Creation
SQL>
SQL> set linesize 120
SQL> TTITLE '** Verify New Agent Details **'
SQL> COL username format a15
SQL> col account_status format a20
SQL> col default_tablespace format a20
SQL> col temporary_tablespace format a20
SQL> col profile format a15
SQL> SELECT username, profile, account_status, default_tablespace, temporary_tablespace, created
  2   FROM dba_users WHERE username='AGENT';

```

Thu Jan 26

** Verify New Agent Details **

page 1

USERNAME	PROFILE	ACCOUNT_STATUS	DEFAULT_TABLESPACE	TEMPORARY_TABLESPACE	CREATED
AGENT	P_SALESAGENT1	OPEN	USERDATA	TEMP	26-JAN-17

```

SQL>
SQL> set linesize 80
SQL>
SQL> --Verify quota space for user
SQL>
SQL> TTITLE '** Quota Report **'
SQL> select tablespace_name, username, bytes USED, max_bytes MAX, (bytes/max_bytes)*100 "%used"
2   from dba_ts_quotas WHERE username='AGENT';

```

Thu Jan 26 page 1
** Quota Report **

TABLESPACE_NAME	USERNAME	USED	MAX	%used
ADHOC	AGENT	0	10485760	0

```

SQL>
SQL> TTITLE OFF
SQL>
SQL> conn agent/Agent28#
Connected.
SQL>
SQL> ttitle 'User Role'
SQL>
SQL> select * from USER_ROLE_PRIVS;


```

Thu Jan 26 page 1
User Role

USERNAME	GRANTED_ROLE	ADM	DEF	OS_
AGENT	R_SALESAGENT1	NO	YES	NO

```
SQL>
SQL>  ttitle 'session privs'
SQL>
SQL>  select * from session_privs;

Thu Jan 26                               page      1
                                         session privs

PRIVILEGE
-----
CREATE SESSION

SQL>
SQL>  -- Test local insert
SQL>
SQL>  ttitle 'Network Test Table Inserts'
SQL>
SQL>  insert into teaadmin.kennynetworktest VALUES (1, 'KennyLocalConnect');

1 row created.

SQL>
SQL>  select * from teaadmin.kennynetworktest;

Thu Jan 26                               page      1
                                         Network Test Table Inserts

REF FIRST_NAME
-----
1 KennyLocalConnect
```

```

SQL>
SQL>
SQL> -- Verify a couple permissions to teaadmin tables
SQL>
SQL> ttitle 'User Select Confirm'
SQL> SELECT count(*) from teaadmin.customer;

Thu Jan 26                               page    1
                                         User Select Confirm

  COUNT(*)
-----
  270

SQL> desc teaadmin.customer;
      Name          Null?    Type
-----  -----
CUST_REF                      NUMBER(3)
FIRST_NAME                     VARCHAR2(20)
LAST_NAME                      VARCHAR2(20)
PREFERRED_AGENT                VARCHAR2(2)

SQL>
SQL> SELECT count(*) from teaadmin.supplier;

Thu Jan 26                               page    1
                                         User Select Confirm

  COUNT(*)
-----
  735

SQL> desc teaadmin.supplier;
      Name          Null?    Type
-----  -----
SUPPLIER_REF                    NUMBER(5)
REGION_REF                      NUMBER(1)
REPRESENTITIVE_CODE             VARCHAR2(30)
COMPANY_NAME                     VARCHAR2(75)
COMMENTS                         VARCHAR2(150)

SQL>
SQL>
SQL> ttitle off
SQL>

```

Network Connectivity Steps

Server Side Instructions

- Create Firewall Rule for Inbound Connections
- Open inbound firewall port Control Panel > Windows Firewall > Advanced > Right click Inbound Rules, click New Rule.
- Choose Protocol TCP, Port 1525 > Click Next > Allow All Connections > Next on Profiles this applies to > Name rule (Oracle 1525) > Finish
- Create Listener
- Run Net Manager (Start > Programs > Oracle - OraDb11g_home1 > Configuration and Migration Tools > Net Manager)
- Expand Local
- Expand Listener
 - Click Green Plus icon
- Type in New Listener Name
- Type in Listener Port (if not default)
- Click Database Services (for the current listener)
- Type in Global Database name (database.domain), Type in Oracle Home folder (C:\app\Administrator\product\11.2.0\dbhome_1)
- Type in SID
- Expand Service Naming
 - Click Database you are making a connection for (TEA)
 - Service Name = Instance
 - Connection Type = Dedicated Server
 - Address Configuration
 - Protocol TCP/IP (or preferred protocol)
 - Host is the machine the database is running on (oralocal)
 - Port Number - 1525 (the port we specified in the Listener, and Inbound Rule above)
- Click Profile
 - Select Naming from drop down list
 - Click Methods tab
 - Add preferred naming methods. 3 methods recommended (Example, TNSNAMES, EZCONNECT, LDAP).
- If Tracing/Logging is required, setup Tracing/Logging details in General drop down menu

To Save

- Click File > Save Network Configuration

Verify Settings

- * Note - File locations below are assuming default file locations
 - These locations can be changed by setting environment variables in Windows.

- Open LISTENER.ORA from C:\app\Administrator\product\11.2.0\dbhome_1\NETWORK\ADMIN
- Check for new entries (examples below, use PORT as an easy identifier)

```
SID_LIST_TRAVELLISTENER =
(SID_LIST =
(SID_DESC =
(GLOBAL_DBNAME = TEA)
(ORACLE_HOME = C:\app\Administrator\product\11.2.0\dbhome_1)
(SID_NAME = TEA)
)
)

TRAVELLISTENER =
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP) (HOST = ICTVM-B6GE6VEST) (PORT = 1525))
)
```

- Open TNSNAMES.ORA from C:\app\Administrator\product\11.2.0\dbhome_1\NETWORK\ADMIN
- Check port, and server mode are correct according to above Net Manager Config

```
TEA =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1525))
)
(CONNECT_DATA =
(SERVER = DEDICATED)
(SERVICE_NAME = TEA)
)
)
```

- Open SQLNET.ORA from C:\app\Administrator\product\11.2.0\dbhome_1\NETWORK\ADMIN
- Check naming methods are correct according to above Net Manager Config
- Can also check Trace and Logging here

```
NAMES DIRECTORY_PATH= (TNSNAMES, EZCONNECT, LDAP)
```

Confirm Listener Settings

- Verify Database is open, services are running, and listener service are started
- Check Windows Services > Start > Run > Services.msc
 - Verify Listener Name is started (Example: OracleOraDb11g_home1TNSListenertravellistener)
 - Verify Database is started (Example: OracleServicetea)

OR

- Use c:\LSNRCTL to start listener
 - C:\LSNRCTL
 - SET CUR LISTENERTRAVEL
 - START

*Can also use RELOAD to restart service

Verify services running with SERVICES command

- Optionally, you can register non default listeners in PFILE

- Add line below to register listener with startup

```
local_listener="(ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1525))"
```

- You can also use the alias for the local_listener, but need to add the listener line from LISTENER.ORA to your TNSNAMES.ORA

Example of listener information added to TNSNAMES below

```
TRAVELLISTENER =
(DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP) (HOST = ICTVM-B6GE6VEST) (PORT = 1525))
)
```

- If desired, create spfile from pfile
- Verify parameter for listener (SQL>show parameter listener)

Test Local Network Connection

```
C:\SQLPLUS agent@tea  
Can use v$session to view connected user(s)  
SELECT sid, osuser, username, server FROM v$session;
```

To Connect to a Remote Server

- Add remote database to TNSNAMES.ORA (example below)

```
TETA =  
(DESCRIPTION =  
  (ADDRESS_LIST =  
    (ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1525))  
  )  
  (CONNECT_DATA =  
    (SERVER = DEDICATED)  
    (SERVICE_NAME = TETA)  
  )  
)
```

- From C:\TNSPING alias

Example -

```
C:\Windows\System32>tnsping max  
TNS Ping Utility for 64-bit Windows: Version 11.2.0.1.0 - Production on 26-JAN-2017 13:42:33  
Copyright (c) 1997, 2010, Oracle. All rights reserved.  
  
Used parameter files:  
C:\app\Administrator\product\11.2.0\dbhome_1\network\admin\sqlnet.ora  
  
Used TNSNAMES adapter to resolve the alias  
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP) (HOST =  
ICTVM-GC04SG8PF) (PORT = 1523))) (CONNECT  
_DATA = (SERVER = DEDICATED) (SERVICE_NAME = travel.charlie)))  
OK (20 msec)
```

- From C:\SQLPLUS username@alias

```
Example C:\SQLPLUS agent@max  
C:\Windows\system32>sqlplus agent@max  
SQL*Plus: Release 11.2.0.1.0 Production on Thu Jan 26 13:41:36 2017  
Copyright (c) 1982, 2010, Oracle. All rights reserved.  
  
Enter password:  
  
Connected to:  
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production  
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

Demonstration/Tests

```
-- Testing Network Connectivity with Teammates
```

```
-- Writes to my own table from others
```

```
SQL> select * from teaadmin.kennynetworktest;
```

```
REF FIRST_NAME
```

```
-----  
1 arthur  
3 sumaya
```

```
SQL>
```

```
-- Writes to Sumaya and Arthur's tables
```

```
-- ARTHUR
```

```
C:\Windows\system32>sqlplus agent@teta
```

```
SQL*Plus: Release 11.2.0.1.0 Production on Thu Jan 26 13:06:26 2017
```

```
Copyright (c) 1982, 2010, Oracle. All rights reserved.
```

```
Enter password:
```

```
Connected to:
```

```
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production  
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

```
SQL> desc teta.arthurnetworktest
```

Name	Null?	Type
ID		NUMBER(3)
FIRSTNAME	NOT NULL	VARCHAR2(15)
LASTNAME	NOT NULL	VARCHAR2(15)

```
SQL> select * from teta.arthurnetworktest
```

```
2 ;
```

```
no rows selected
```

```
SQL> insert into teta.arthurnetworktest values (1, 'Kenny', 'Vigar');
```

```
1 row created.
```

```
SQL> select * from teta.arthurnetworktest  
2 ;
```

ID	FIRSTNAME	LASTNAME
1	Kenny	Vigar

```
SQL>
```

```
-- SUMAYA
```

```
C:\Windows\system32>sqlplus agent@sumaya
```

```
SQL*Plus: Release 11.2.0.1.0 Production on Thu Jan 26 13:12:06 2017
```

```
Copyright (c) 1982, 2010, Oracle. All rights reserved.
```

```
Enter password:
```

```
Connected to:
```

```
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production  
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

```
SQL> desc travel.sumaya_networktest
```

```
ERROR:
```

```
ORA-04043: object travel.sumaya_networktest does not exist
```

```
SQL> desc tea.sumaya_networktest
```

Name	Null?	Type
ID	NOT NULL	NUMBER
NAME		VARCHAR2 (20)

```
SQL> insert into tea.sumaya_networktest values (3, 'Kenny');
```

```
1 row created.
```

```
SQL> select * from tea.sumaya_networktest;
```

ID	NAME
3	Kenny

```
SQL> commit;
```

```
Commit complete.
```

```
-- Write to Max table
```

```
C:\Windows\system32>sqlplus agent@max

SQL*Plus: Release 11.2.0.1.0 Production on Thu Jan 26 13:28:53 2017

Copyright (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> desc traveldba.maxnetworktest;
Name          Null?    Type
-----        -----    -----
STUDENT_ID      NOT NULL NUMBER(2)
FIRST_NAME       VARCHAR2(30)
LAST_NAME        VARCHAR2(30)

SQL> insert into traveldba.maxnetworktest (first_name, last_name) values ('kenny', 'vigar');

1 row created.

SQL> select * from traveldba.maxnetworktest;

STUDENT_ID FIRST_NAME           LAST_NAME
-----        -----           -----
1 kenny                 vigar

SQL> commit;

Commit complete.
```

tnsnames.ora

```
# tnsnames.ora Network Configuration File: C:\app\Administrator\product\11.2.0\dbhome_1\NETWORK\ADMIN\tnsnames.ora
# Generated by Oracle configuration tools.

SUMAYA =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP) (HOST = ICTVM-C1OR4PMV0) (PORT = 1525))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = TRAVEL)
    )
  )

MAX =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP) (HOST = ICTVM-GC04SG8PF) (PORT = 1523))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = travel.charlie)
    )
  )

TEA =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1525))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = TEA)
    )
  )

TRAVELLISTENER =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP) (HOST = ICTVM-B6GE6VEST) (PORT = 1525))
  )

LISTENERALT =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1523))
  )
```

```

ORACL_CONNECTION_DATA =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = IPC) (KEY = EXTPROC1521))
)
(CONNECT_DATA =
(SID = CLRExtProc)
(PRESENTATION = RO)
)
)

ORANT11G =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1521))
)
(CONNECT_DATA =
(SERVER = DEDICATED)
(SERVICE_NAME = orant11g)
)
)

APEXDB =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1521))
)
(CONNECT_DATA =
(SERVER = DEDICATED)
(SERVICE_NAME = apexdb)
)
)

LISTENER_APEXDB =
(ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1521))

TEATEST =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1523))
)
(CONNECT_DATA =
(SERVER = DEDICATED)
(SERVICE_NAME = TEATEST)
)
)

```

listener.ora

```
# listener.ora Network Configuration File: C:\app\Administrator\product\11.2.0\dbhome_1\NETWORK\ADMIN\listener.ora
# Generated by Oracle configuration tools.

SID_LIST_TRAVELLISTENER =
  (SID_LIST =
    (SID_DESC =
      (GLOBAL_DBNAME = TEA)
      (ORACLE_HOME = C:\app\Administrator\product\11.2.0\dbhome_1)
      (SID_NAME = TEA)
    )
  )

SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = CLRExtProc)
      (ORACLE_HOME = C:\app\Administrator\product\11.2.0\dbhome_1)
      (PROGRAM = extproc)
      (ENVS = "EXTPROC_DLLS=ONLY:C:\app\Administrator\product\11.2.0\dbhome_1\bin\oraclr11.dll")
    )
  )

LISTENERALT =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1523))
  )

TRAVELLISTENER =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP) (HOST = ICTVM-B6GE6VEST) (PORT = 1525))
  )

ADR_BASE_TRAVELLISTENER = C:\app\Administrator\product\11.2.0\dbhome_1\log

LISTENER =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP) (HOST = oralocal) (PORT = 1521))
  )

ADR_BASE_LISTENER = c:\TEA\listner.log

SID_LIST_LISTENERALT =
  (SID_LIST =
    (SID_DESC =
      (GLOBAL_DBNAME = tea)
      (ORACLE_HOME = C:\app\Administrator\product\11.2.0\dbhome_1)
      (SID_NAME = tea)
    )
    (SID_DESC =
      (GLOBAL_DBNAME = teatest)
      (ORACLE_HOME = C:\app\Administrator\product\11.2.0\dbhome_1)
```

```

        (SID_NAME = teatest)
    )
(SID_DESC =
(GLOBAL_DBNAME = Oracle8)
(SID_NAME = ORCL)
)
)

ADR_BASE_LISTENERALT = C:\app\Administrator
TRACE_LEVEL_LISTENER = USER

```

sqlnet.ora

```

# sqlnet.ora Network Configuration File: C:\app\Administrator\product\11.2.0\dbhome_1\NETWORK\ADMIN\sqlnet.ora
# Generated by Oracle configuration tools.

# This file is actually generated by netca. But if customers choose to
# install "Software Only", this file wont exist and without the native
# authentication, they will not be able to connect to the database on NT.

SQLNET.AUTHENTICATION_SERVICES= (NTS)

TRACE_LEVEL_CLIENT = USER

NAMES.DIRECTORY_PATH= (TNSNAMES, EZCONNECT, LDAP)

TRACE_LEVEL_SERVER = USER

TRACE_DIRECTORY_CLIENT = C:\TEMP\trace

TRACE_FILE_CLIENT = clinettrace.log

LOG_FILE_CLIENT = clientlogging.log

TRACE_DIRECTORY_SERVER = C:\TEMP\trace

TRACE_FILE_SERVER = servertrace.log

LOG_DIRECTORY_CLIENT = C:\TEMP\trace

LOG_DIRECTORY_SERVER = C:\TEMP\trace

DIAG_ADR_ENABLED = OFF

```


Data Warehouse Code Supplement

March 11th, 2017

Kenny Vigar
kvigar@gmail.com

Strategy Plan	Page
Background	1
External Data Load	
- External Table	2
- SQLLDR	3
Fact Table Partitioning	3
Create Fact Table	4
Merge Data into Fact Table	5
Tuning Fact Table - Outline/Materialized View	6
Create Dimension Table	7
Analysis of Data – Excel Pivots and Graphs	13
Analysis of Data –SQLPLUS SUMSUM Reporting	16
Analysis and Marketing Recommendations	17

Background

Travel Experts Travel Agency can benefit from a data warehouse database design by allowing information stored in the database to be accessed quicker than a normalized database design. Using a Fact Table in our warehouse as the primary table, information can be queried as a whole, or data can be accessed in a warehouse Dimension which gives a quick accurate summary of *pre-calculated totals* without any additional SQL knowledge for management or analysts.

External Data Load

```
-- Part 1 a)

-- Load Data into Database Script
-----

-- External Table Load
-----

-- Prep create dump file to import to database.
-- This demonstration exports the adhoc_allsales table to a dumpfile, which will use an External Table
-- to load it back into the 'Data Warehouse' database.

-- Verify Oracle Directory is setup

SELECT * from all_directories;

OWNER          DIRECTORY_NAME          DIRECTORY_PATH
-----          -----
SYS            DPDIR                 c:\temp\teadp
SYS            TEMPDIR               c:\tea\tempdir

CREATE TABLE ext_table_all_sales
ORGANIZATION External (type oracle datapump
    default directory dppdir
    location ('_sales_dump.dmp')) AS SELECT * from adhoc_allsales;

-- Verify dump file was created

C:\TEA\tempdir>dir *.dmp
Volume in drive C has no label.
Volume Serial Number is D4CA-D3D5

Directory of C:\TEA\tempdir

11/03/2017  12:29 PM      253,952 _SALES_DUMP.DMP
              2 File(s)        352,256 bytes
              0 Dir(s)   12,691,902,464 bytes free

-- Create Table with External Table (and dump file from above)

CREATE TABLE Temp_Sales_Data_Load
(saledate           DATE,
 cust_id            NUMBER(3),
 itinerary_num      NUMBER(5),
 agent               VARCHAR2(2),
 booking_num        VARCHAR2(10),
 product_cat        NUMBER(3),
 supplier_id        NUMBER(5),
 supp_office        NUMBER(1),
 trip_start         DATE,
 trip_end           DATE,
 travelclass        VARCHAR2(5),
 traveller_count    NUMBER(2),
 product             VARCHAR2(30),
 product_des         VARCHAR2(40),
```

```

destination          VARCHAR2(30),
dest_id              VARCHAR2(10),
credit_card          VARCHAR2(10),
credit_card_exp      DATE,
credit_card_num      NUMBER(20),
bill_date            DATE,
bill_des             VARCHAR2(15),
baseprice            NUMBER(10),
totalpricewtax      NUMBER(10),
billedamt           NUMBER(10),
agencyfee            VARCHAR2(15),
feeamt               NUMBER(10),
agencycomm           NUMBER(10),
payment_ref          NUMBER(10),
commission_ref       NUMBER(10))

ORGANIZATION EXTERNAL
  (TYPE ORACLE_DATAPUMP
  DEFAULT DIRECTORY dpdir
  LOCATION ('_sales_dump.dmp'));

-- Verify data load

SQL> select count(*) from temp_sales_data_load;

COUNT(*)
-----
1341

-----

-- Method 2
-- SQL LDR
-----

-- SQLLDR can also be used to import data to the database.
-- Required files are a csv file of data, SQLLDR control file and a SQLLDR log file
-- Control File is attached, but copied below as well

-- Control File - sales.csv
options(SKIP=1, direct=y)
LOAD DATA
INFILE 'C:\TEA\csv\sales.csv'
  truncate INTO TABLE adhoc_allsales
FIELDS TERMINATED BY "," optionally enclosed by """
TRAILING NULLCOLS
(saledate DATE 'DD/MM/YYYY',
cust_id integer external,
itinerary_num integer external,
agent char,
booking_num char,
product_cat integer external,
supplier_id integer external,
Supp_office integer external,
trip_start DATE 'DD-MM-YYYY',
trip_end DATE 'DD/MM/YYYY',
travelclass char,
traveller_count integer external,
product char,
product_des char,
destination char,
dest_id char,
credit_card char,
credit_card_exp DATE 'DD/MM/YYYY',
credit_card_num integer external,
bill_date DATE 'DD/MM/YYYY',
bill_des char,
baseprice integer external,
totalpricewtax integer external,
billedamt integer external,
agencyfee integer external,

```

```

feeamt integer external,
agencycomm integer external)

-- After control file creation, use command below to load data

C:\SQLLDR teaadmin/teaadmin1234 control=C:\tea\csv\control\adhocsalescontrol.ctl
log=C:\tea\csv\control\controllogs\adhocsalescontrol.log
bad=C:\tea\csv\control\controllogs\badadhocsales.log

```

Fact Table code includes - Fact Table tablespace partitioning

- Fact Table creation and merge statement
- Fact Table material view
- Fact Table Tuning
 - bitmap index
 - Fact Table outline

```

-----  

-- Part 1 b and d)  

-- Prepare for Fact Table Load  

-- create partitions as part of Fact Table Tuning  

-----  

-- Create Fact Table Partitions  

create tablespace dw_fact_partition_1 datafile 'C:\tea\disk01\partition1\dw_fact_partition_1.dbf'  

  size 10m EXTENT MANAGEMENT local UNIFORM SIZE 200k;  

create tablespace dw_fact_partition_2 datafile 'C:\tea\disk02\partition2\dw_fact_partition_2.dbf'  

  size 10m EXTENT MANAGEMENT local UNIFORM SIZE 200k;  

create tablespace dw_fact_partition_3 datafile 'C:\tea\disk03\partition3\dw_fact_partition_3.dbf'  

  size 10m EXTENT MANAGEMENT local UNIFORM SIZE 200k;  

create tablespace dw_fact_partition_4 datafile 'C:\tea\disk04\partition4\dw_fact_partition_4.dbf'  

  size 10m EXTENT MANAGEMENT local UNIFORM SIZE 200k;  

create tablespace dw_fact_partition_5 datafile 'C:\tea\disk05\partition5\dw_fact_partition_5.dbf'  

  size 10m EXTENT MANAGEMENT local UNIFORM SIZE 200k;  

-- Verify Tablespace Creation  

set linesize 100  

set head on  

COL "Table Space Name" format A20  

COL "Size in Bytes" format 99999999999  

COL Location format A50  

SELECT ts.name "Table Space Name", df.name Location, ((df.bytes/1024)/1024) "MB"  

FROM v$logfile df JOIN v$tablespace ts ON ts.ts# = df.ts#  

WHERE ts.name like 'DW_FACT_PART%';  

Table Space Name          LOCATION          MB  

-----  

DW_FACT_PARTITION_1    C:\TEA\DISK01\PARTITION1\DW_FACT_PARTITION_1.DBF    10  

DW_FACT_PARTITION_2    C:\TEA\DISK02\PARTITION2\DW_FACT_PARTITION_2.DBF    10  

DW_FACT_PARTITION_3    C:\TEA\DISK03\PARTITION3\DW_FACT_PARTITION_3.DBF    10  

DW_FACT_PARTITION_4    C:\TEA\DISK04\PARTITION4\DW_FACT_PARTITION_4.DBF    10  

DW_FACT_PARTITION_5    C:\TEA\DISK05\PARTITION5\DW_FACT_PARTITION_5.DBF    10  

ALTER USER teaadmin QUOTA 10M ON DW_FACT_PARTITION_1;  

ALTER USER teaadmin QUOTA 10M ON DW_FACT_PARTITION_2;  

ALTER USER teaadmin QUOTA 10M ON DW_FACT_PARTITION_3;  

ALTER USER teaadmin QUOTA 10M ON DW_FACT_PARTITION_4;  

ALTER USER teaadmin QUOTA 10M ON DW_FACT_PARTITION_5;

```

```
-- Create Sequence for Fact Table

CREATE SEQUENCE fact_table_seq
START WITH      99
INCREMENT BY   1
NOCACHE
NOCYCLE;

-- Verify Sequence was Created

SQL> select * from user_sequences where sequence_name like 'FACT%';

SEQUENCE_NAME          MIN_VALUE  MAX_VALUE INCREMENT_BY C O CACHE_SIZE LAST_NUMBER
-----  -----  -----  -----  -----  -----  -----
FACT_TABLE_SEQ           1 1.0000E+28           1 N N          0          99

-- Create Fact Table
-- - Partition over tablespaces

create table FACT_TRAVEL
(fact_table_seq number(6) ,
customer_name varchar2(20), -- concat the insert
province_name varchar2(2),
city_name varchar2(10),
phone number number(12),
business_number number(12),
address varchar2(35),
travel_des varchar2(20),
product_category number(5),
product_name varchar2(30),
booking_ref varchar2(10),
itinerary_ref number(5),
supplier_name varchar2(50),
base_price number(10,2),
fee_des varchar(30),
fee_amt number (10,2),
bill_total number(10,2),
bill_date date,
bill_year number(4),
bill_month number(2),
bill_day number(2),
payment_ref number(5),
cc_type varchar2(10),
CONSTRAINT fact_table_seq_pk PRIMARY KEY (Fact Table Seq) USING INDEX
(CREATE INDEX fact_table_idx ON teaadmin.FACT_TRAVEL(fact_table_seq)
PCTFREE 10 INITTRANS 2
STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
TABLESPACE indx)) partition by range(fact table seq)
(partition part1 values less than (50) tablespace DW_FACT_PARTITION_1,
partition part2 values less than (100) tablespace DW_FACT_PARTITION_2,
partition part3 values less than (150) tablespace DW_FACT_PARTITION_3,
partition part4 values less than (200) tablespace DW_FACT_PARTITION_4,
partition part5 values less than (maxvalue) tablespace DW_FACT_PARTITION_5);
comment on table FACT_TRAVEL is 'Travel Experts Travel Agency - Data Warehouse';
```

```

-- Merge into Fact

merge into FACT_TRAVEL facttable
using (select to_char(c.first_name||' '||c.last_name) as customer_name, cd.prov, cd.city,
cd.home_phone, cd.business_phone,
        cd.address, tc.class_des, p.product_cat, p.product_des, i.booking_ref, i.itinerary_ref,
sup.company_name, pay.base_price, f.fee_des,
        pay.fee_amt, pay.payment_total, pay.bill_date, to_number(to_char(pay.bill_date, 'YYYY')) year,
to_number(to_char(pay.bill_date, 'MM')) month,
        to number(to char(pay.bill date, 'dd')) day, pay.payment ref, creditc.cc_type
        from customer c join customerdetail cd on c.cust_ref=cd.cust_ref
        join sale s on c.cust_ref=s.cust_ref
        join itinerary i on i.payment_ref = s.payment_ref and i.itinerary_ref = s.itinerary_ref
        join travelclass tc on tc.travel_class = i.travel_class
        join product p on p.product cat = i.product cat and p.supplier_ref=i.supplier_ref and
                    p.region_ref=i.region_ref
        join supplier sup on sup.supplier ref = p.supplier_ref and sup.region_ref=p.region_ref
        join payment pay on pay.payment_ref=s.payment_ref
        join fee f on f.fee_ref=pay.fee_ref
        join cc creditc on creditc.cust_ref = c.cust_ref) factsquery
ON (facttable.customer_name = factsquery.customer_name and
facttable.province_name = factsquery.prov and
facttable.city_name = factsquery.city and
facttable.phone_number = factsquery.home_phone and
facttable.business_number = factsquery.business_phone and
facttable.address = factsquery.address and
facttable.travel_des = factsquery.class_des and
facttable.product_category = factsquery.product_cat and
facttable.product_name = factsquery.product_des and
facttable.booking_ref = factsquery.booking_ref and
facttable.itinerary_ref = factsquery.itinerary_ref and
facttable.supplier_name = factsquery.company_name and
facttable.base_price = factsquery.base_price and
facttable.fee_des = factsquery.fee_des and
facttable.fee_amt = factsquery.fee_amt and
facttable.bill_total = factsquery.payment_total and
facttable.bill_date = factsquery.bill_date and
facttable.bill_year = factsquery.year and
facttable.bill_month = factsquery.month and
facttable.bill_day = factsquery.day and
facttable.payment_ref = factsquery.payment_ref and
facttable.cc_type = factsquery.cc_type)
when not matched then insert values (fact_table_seq.nextval,
                                    factsquery.customer_name,
                                    factsquery.prov,
                                    factsquery.city,
                                    factsquery.home_phone,
                                    factsquery.business_phone,
                                    factsquery.address,
                                    factsquery.class_des,
                                    factsquery.product_cat,
                                    factsquery.product_des,
                                    factsquery.booking_ref,
                                    factsquery.itinerary_ref,
                                    factsquery.company_name,
                                    factsquery.base_price,
                                    factsquery.fee_des,
                                    factsquery.fee_amt,
                                    factsquery.payment_total,
                                    factsquery.bill_date,
                                    factsquery.year,
                                    factsquery.month,
                                    factsquery.day,
                                    factsquery.payment_ref,
                                    factsquery.cc_type);

```

1109 rows merged.

```

-- Further tuning Data Warehouse

-- need to use 'local' option as this is a partitioned table

create bitmap index fact_cust_name_idx on fact_travel(customer_name) local;
create bitmap index fact_prov_name_idx on fact_travel(province_name) local;
create bitmap index fact_city_name_idx on fact_travel(city_name) local;
create bitmap index fact_travel_des_idx on fact_travel(travel_des) local;
create bitmap index fact_product_cat_idx on fact_travel(product_category) local;
create bitmap index fact_product_name_idx on fact_travel(product_name) local;
create bitmap index fact_booking_idx on fact_travel(booking_ref) local;
create bitmap index fact_itinerary_idx on fact_travel(itinerary_ref) local;
create bitmap index fact_supplier_name_idx on fact_travel(supplier_name) local;
create bitmap index fact_fee_description_idx on fact_travel(fee_des) local;
create bitmap index fact_payment_ref_idx on fact_travel(payment_ref) local;
create bitmap index month_idx on fact_travel(bill_month) local;
create bitmap index day_idx on fact_travel(bill_day) local;
create bitmap index year_idx on fact_travel(bill_year) local;

-- Further Fact Table Tuning!
-- Create Outline on Fact_Travel

create outline fact_travel ON
select to_char(c.first_name||' '||c.last_name) as customer_name, cd.prov, cd.city, cd.home_phone,
cd.business_phone, cd.address, tc.class_des, p.product_cat, p.product_des, i.booking_ref,
i.itinerary_ref, sup.company_name, pay.base_price, f.fee_des, pay.fee_amt, pay.total, pay.bill_date,
to_number(to_char(pay.bill_date, 'YYYY')) year, to_number(to_char(pay.bill_date, 'MM')) month,
to_number(to_char(pay.bill_date, 'dd')) day, pay.payment_ref, creditc.cc_type
  from customer c join customerdetail cd on c.cust_ref=cd.cust_ref
    join sale s on c.cust_ref=s.cust_ref
      join itinerary i on i.payment_ref = s.payment_ref and i.itinerary_ref =
        s.itinerary_ref
      join travelclass tc on tc.travel_class = i.travel_class
      join product p on p.product_cat = i.product_cat and p.supplier_ref=i.supplier_ref
        and p.region_ref=i.region_ref
      join supplier sup on sup.supplier_ref = p.supplier_ref
        and sup.region_ref=p.region_ref
      join payment pay on pay.payment_ref=s.payment_ref
      join fee f on f.fee_ref=pay.fee_ref
      join cc creditc on creditc.cust_ref = c.cust_ref;

Outline created.

-- Weekly updated Material View +7

create materialized view fact_travel_mt_vw
  refresh force with rowid
  start with sysdate next sysdate +7
  as select to_char(c.first_name||' '||c.last_name) as customer_name, cd.prov, cd.city,
  cd.home_phone, cd.business_phone,
  cd.address, tc.class_des, p.product_cat, p.product_des, i.booking_ref, i.itinerary_ref,
  sup.company_name, pay.base_price, f.fee_des,
  pay.fee_amt, pay.total, pay.bill_date, to_number(to_char(pay.bill_date, 'YYYY')) year,
  to_number(to_char(pay.bill_date, 'MM')) month, to_number(to_char(pay.bill_date, 'dd')) day,
  pay.payment_ref, creditc.cc_type
  from customer c join customerdetail cd on c.cust_ref=cd.cust_ref
    join sale s on c.cust_ref=s.cust_ref
    join itinerary i on i.payment_ref = s.payment_ref and i.itinerary_ref = s.itinerary_ref
    join travelclass tc on tc.travel_class = i.travel_class
    join product p on p.product_cat = i.product_cat and p.supplier_ref=i.supplier_ref
      and p.region_ref=i.region_ref
    join supplier sup on sup.supplier_ref = p.supplier_ref and sup.region_ref=p.region_ref
    join payment pay on pay.payment_ref=s.payment_ref
    join fee f on f.fee_ref=pay.fee_ref
    join cc creditc on creditc.cust_ref = c.cust_ref;

comment on materialized view fact_travel_mt_vw is 'Fact Table Materialized View';

```

Materialized view created.

Dimension Table code includes -

- Dimension table creation
- merge with pre calculated total queries
- material views with pre calculated total queries

```
-- Part 1c)

-- Create Dimension Tables
-----

-- Create Product Total Dimension Table
-----


create table teaadmin.dim_product_total
(product varchar2(25),
 total_sales number(10,2),
 CONSTRAINT prod_total_pk PRIMARY KEY (product) USING INDEX
        (CREATE INDEX prod_total_indx ON teaadmin.dim_product_total(product)
          PCTFREE 10 INITTRANS 2
          STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
          TABLESPACE indx) );
comment on table dim_product_total is 'Dimension Table for Product Totals';

-- Query for Confirming and Loading Data

select i.product, to_char(sum(p.total), '$9,999,999.99') as Total_Sales
from itinerary i join sale s on i.itinerary_ref = s.itinerary_ref
    join payment p on s.payment_ref=p.payment_ref
    group by product;

-- Merge Command to load Data to Product Total Dimension

merge into dim_product_total dtotal
using (select i.product, sum(p.total)as salessum
       from itinerary i join sale s on i.itinerary_ref = s.itinerary_ref
       join payment p on s.payment_ref=p.payment_ref
       group by product) products
ON (dtotal.product = products.product)
when not matched then insert values (products.product, products.salessum)
when matched then update set total_sales=total_sales+salessum;

-- Select data from Product Total Dimension Table

select d_pt.product,to_char(sum(d_pt.total_sales), '$9,999,999.99') Sale_Sum
from dim_product_total d_pt
group by d_pt.product;

-- For extra practice, trying a Material View for same product total information

-- Material View with Weekly Information

create materialized view dim_product_view
refresh force with rowid
start with sysdate next sysdate +7
as
select i.product, sum(p.total)as salessum
       from itinerary i join sale s on i.itinerary_ref = s.itinerary_ref
       join payment p on s.payment_ref=p.payment_ref
       group by product;

comment on materialized view dim_product_view is 'Dimension Product Total Materialized View';
```

```

-- Select from Material View

select d_pv.product,to_char(sum(d_pv.salessum), '$9,999,999.99') Sale_Sum
from dim_product_view d_pv
group by d_pv.product;

-----
-- Create City Total Dimension Table
-----

create table teaadmin.dim_city_total
(city varchar2(25),
 total sales number(10,2),
 CONSTRAINT city_total_pk PRIMARY KEY (city) USING INDEX
     (CREATE INDEX city total indx ON teaadmin.dim_city_total(city)
      PCTFREE 10 INITTRANS 2
      STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
      TABLESPACE indx) );
comment on table dim city total is 'Dimension Table for City Totals';

-- Query for Confirming and Loading Data

select cd.city, sum(p.total) as City_Sales
from customerdetail cd join customer c on cd.cust_ref=c.cust_ref
    join sale s on c.cust_ref = s.cust_ref
        join payment p on s.payment_ref = p.payment_ref
            group by city;

-- Merge Command to load Data to City Total Dimension

merge into dim_city_total ctotal
using (select cd.city, sum(p.total) as citysum
       from customerdetail cd join customer c on cd.cust_ref=c.cust_ref
           join sale s on c.cust_ref = s.cust_ref
               join payment p on s.payment_ref = p.payment_ref
                   group by city) citytotal
ON (ctotal.city = citytotal.city)
when not matched then insert values (citytotal.city, citytotal.citysum)
when matched then update set total_sales=total_sales+citysum;

-- Select data from City Total Dimension Table

select d_ct.city,to_char(sum(d_ct.total_sales), '$9,999,999.99') Sale_Sum
from dim_city_total d_ct
group by d_ct.city;

-- For extra practice, trying a Material View for same city total information

-- Material View with Weekly Information

create materialized view dim_city_view
refresh force with rowid
start with sysdate next sysdate +7
as select cd.city, sum(p.total) as City_Sales
from customerdetail cd join customer c on cd.cust_ref=c.cust_ref
    join sale s on c.cust_ref = s.cust_ref
        join payment p on s.payment_ref = p.payment_ref
            group by city;

comment on materialized view dim_city_view is 'Dimension City Total Materialized View';

```

```

-----  

-- Create Travel Class Total Dimension Table  

-----  

create table teaadmin.dim_travel_class_total  

(travel_class varchar2(25),  

 total_count number(10,2),  

CONSTRAINT tc_total_pk PRIMARY KEY (travel_class) USING INDEX  

    (CREATE INDEX tc_total_indx ON teaadmin.dim_travel_class_total(travel_class)  

        PCTFREE 10 INITTRANS 2  

        STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)  

        TABLESPACE indx) );  

comment on table dim_travel_class_total is 'Dimension Table for Travel Class Totals';  

-- Query for Confirming and Loading Data  

select tc.class_des, count(i.travel_class) as ClassCount  

from itinerary i join travelclass tc on tc.travel_class = i.travel_class  

where i.travel_class is not null  

group by tc.class_des  

order by count(i.travel_class) desc;  

-- Merge Command to load Data to Travel Class Total Dimension  

merge into dim travel class total tctotal  

using (select tc.class_des, count(i.travel_class) as ClassCount  

from itinerary i join travelclass tc on tc.travel_class = i.travel_class  

where i.travel_class is not null  

group by tc.class_des ) travelclasstotal  

ON (tctotal.travel_class = travelclasstotal.class_des)  

when not matched then insert values (travelclasstotal.class_des, travelclasstotal.ClassCount)  

when matched then update set total_count=total_count+ClassCount;  

-- Select data from Travel Class Total Dimension Table  

select d_tct.travel_class,sum(d_tct.total_count) Travel_Count_Sum  

from dim_travel_class_total d_tct  

group by d_tct.travel_class  

order by Travel_Count_Sum desc;  

-- For extra practice, trying a Material View for same travel count total information  

-- Material View with Weekly Information  

create materialized view dim_travel_count_view  

refresh force with rowid  

start with sysdate next sysdate +7  

as select d_tct.travel_class,sum(d_tct.total_count) Travel_Count_Sum  

from dim_travel_class_total d_tct  

group by d_tct.travel_class  

order by Travel_Count_Sum desc;  

comment on materialized view dim_travel_count_view is 'Dimension Travel Count Materialized View';  

-- Select data from Material View  

select * from dim_travel_count_view order by travel_count_sum desc;

```

```

-----
-- Sales by Date Summary
-----

-- Create Date Total Dimension Table

create table teaadmin.dim_date_total
(year number(4),
month number(2),
day number (2),
total_sum number(10,2),
CONSTRAINT date_total_pk PRIMARY KEY (year,month,day) USING INDEX
    (CREATE INDEX date total indx ON teaadmin.dim_date_total(year, month,day)
        PCTFREE 10 INITTRANS 2
        STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
        TABLESPACE indx) );
comment on table dim_date_total is 'Dimension Table for Bill Date Sales Totals';

-- Query for Confirming and Loading Data

select extract (year from bill_date) yy,
       extract (month from bill_date) mon,
       extract (day from bill_date) dd,
       to_char(sum(billedamt), '$9,999,999.99') total
  from adhoc_allsales
 group by (bill date)
 order by yy,mon,dd;

-- Unable to use Merge with Extract above.... creating a temporary view to select data from to merge to
-- dimension table. Thanks for the tip Randall.

-- Temp View Creation

create or replace view temp_extract_date
as select extract (year from bill date) yy,
           extract (month from bill_date) mon,
           extract (day from bill date) dd,
           sum(billedamt) total
      from adhoc_allsales
     group by (bill_date)
    order by yy,mon,dd;

-- Merge Command to load Data to Date Total Dimension

merge into dim_date_total dtot
using (select * from temp_extract_date) datetot
  ON (dtot.year = datetot.yy and dtot.month = datetot.mon and dtot.day = datetot.dd)

    when not matched then insert values (datetot.yy, datetot.mon, datetot.dd,
datetot.total)
    when matched then update set total_sum=total_sum+total;

-- For extra practice, trying a Material View for same Date total information

-- Material View with Weekly Information

create materialized view dim_date_view
refresh force with rowid
start with sysdate next sysdate +7
as select yy,mon,dd,to_char(sum(total), '$9,999,999.99') total from temp_extract_date
group by yy,mon,dd
order by yy,mon;

comment on materialized view dim_date_view is 'Dimension Date Total Materialized View';

```

```

-----
-- Customer Total
-----

-- Create Customer Total Dimension Table

create table teaadmin.dim_cust_total
(customer_name varchar2(25),
 total_sales number(10,2),
 CONSTRAINT customer_total_pk PRIMARY KEY (customer_name) USING INDEX
     (CREATE INDEX customer_total_indx ON teaadmin.dim_cust_total(customer_name)
      PCTFREE 10 INITTRANS 2
      STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5)
      TABLESPACE indx) );
comment on table dim_cust_total is 'Dimension Table for Customer Totals';

-- Query for Confirming and Loading Data

select to_char(c.first_name||' '||c.last_name) as customer_name, to_char(sum(p.total), '$9,999,999.99')
as customer_sales
from customer c join sale s on c.cust_ref = s.cust_ref
join payment p on s.payment_ref = p.payment_ref
group by c.first_name||' '||c.last_name
order by c.first_name||' '||c.last_name;

-- Merge Command to load Data to Customer Total Dimension

merge into dim_cust_total custtotal
using (select to_char(c.first_name||' '||c.last_name) as customer_name, sum(p.total) as customer_sales
from customer c join sale s on c.cust_ref = s.cust_ref
join payment p on s.payment_ref = p.payment_ref
group by c.first_name||' '||c.last_name
order by c.first_name||' '||c.last_name) customertotal
ON (custtotal.customer_name = customertotal.customer_name)

when not matched then insert values (customertotal.customer_name,
customertotal.customer_sales)
when matched then update set total_sales=total_sales+customer_sales;

-- For extra practice, trying a Material View for same customer total information

-- Material View with Weekly Information

create materialized view dim_cust_view
refresh force with rowid
start with sysdate next sysdate +7
as select to_char(c.first_name||' '||c.last_name) as customer_name, sum(p.total) as
customer_sales
from customer c join sale s on c.cust_ref = s.cust_ref
join payment p on s.payment_ref = p.payment_ref
group by c.first_name||' '||c.last_name
order by c.first_name||' '||c.last_name;

comment on materialized view dim_cust_view is 'Dimension Customer Total Materialized View';

```

Data Mining –

```
-- Part 2 a)

-- Data Mining
-- Excel Pivot Chart
-----

-- Export Fact Table CSV for Excel data analysis

-- Verify Oracle Directory is setup

SELECT * from all_directories;

OWNER          DIRECTORY_NAME          DIRECTORY_PATH
-----          -----          -----
SYS            DPDIR                c:\temp\teadp
SYS            TEMPDIR              c:\tea\tempdir

-- Generate CSV file

create or replace procedure Travel_Experts_Fact_Export
as
  x sys.utl_file.file_type;
begin
  x:= sys.utl_file.fopen('DPDIR', 'Travel Experts_Fact_Export.csv', 'w');
  for y in (select * from fact_travel) loop
    sys.utl_file.put_line(x, y.fact_table_seq||','||y.customer_name||','||y.province_name
||','||y.city_name||','||y.phone_number||','||y.business_number||','||y.address||','||y.travel_des||','||
y.product_category||','||y.product_name||','||y.booking_ref||','||y.itinerary_ref||','||y.supplier_name||',
'|| y.base price||','||y.fee des||','||y.fee amt||','||y.bill total||','||y.bill date||',
'|| y.bill_year ||','||y.bill_month||','||y.bill_day||','||y.payment_ref||','||y.cc_type);
  end loop;
  sys.utl_filefclose(x);
end;
/

exec Travel_Experts_Fact_Export;

-- Verify dump file was created

C:\TEMP\teadp>dir *.csv
Volume in drive C has no label.
Volume Serial Number is D4CA-D3D5

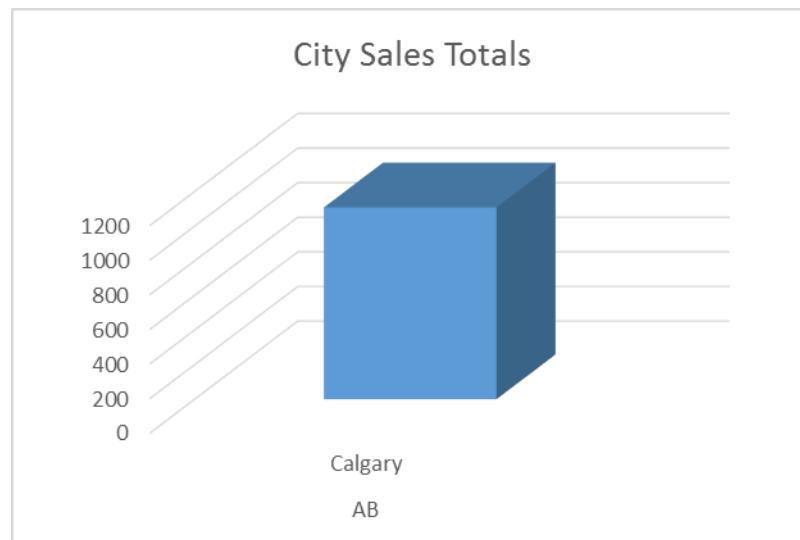
Directory of C:\TEMP\teadp

18/03/2017  12:39 AM           203,174 Travel_Experts_Fact_Export.csv
               1 File(s)        203,174 bytes
               0 Dir(s)   7,208,058,880 bytes free
```

-- Importing to Excel

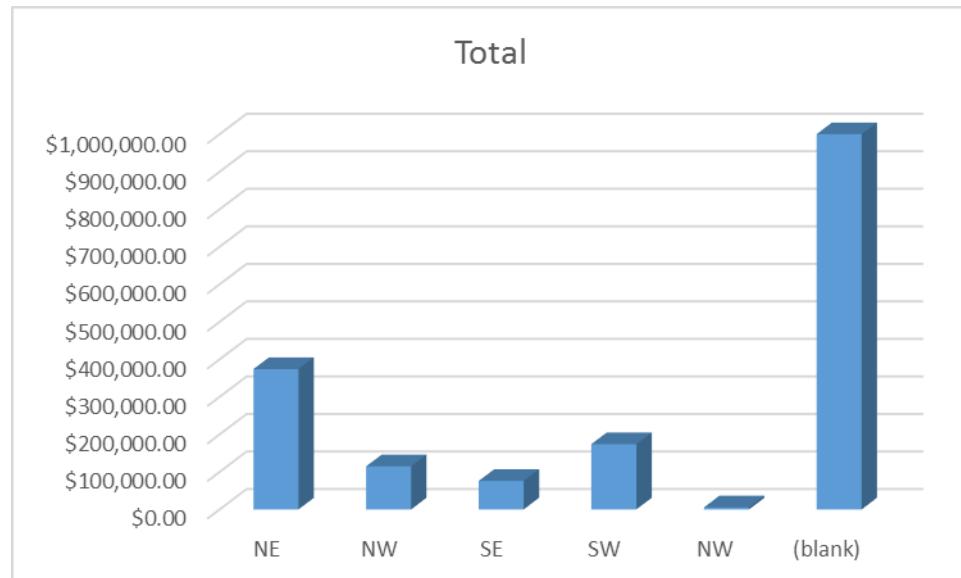
Customer Totals by Province/City

Row Labels	Customer Count
AB	1109
Calgary	1109
Grand Total	1109



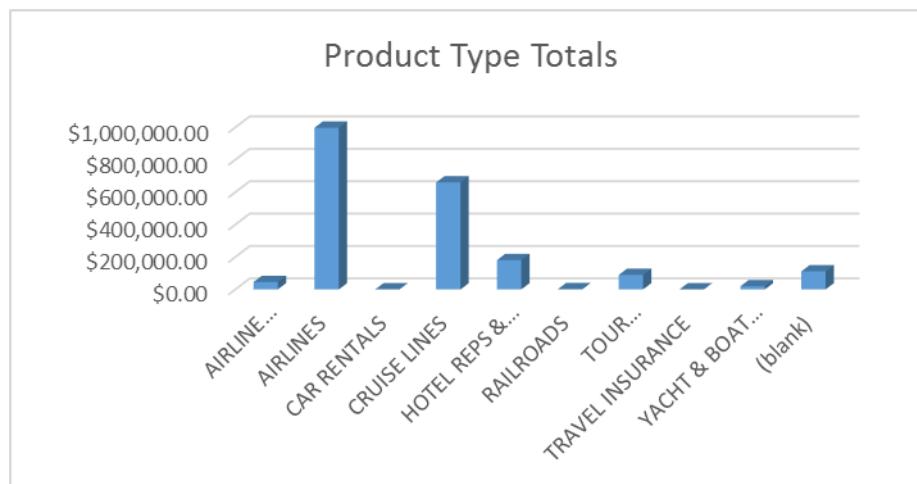
Sales by Region

Customer Region	Sales by Region
NE	\$374,064.00
NW	\$114,794.00
SE	\$76,091.00
SW	\$173,902.00
NW	\$5,600.00
(blank)	\$1,365,286.00
Grand Total	\$2,109,737.00



Product Total Sales

Product Category	Total Sales
AIRLINE CONSOLIDATORS	\$45,389.00
AIRLINES	\$997,732.00
CAR RENTALS	\$800.00
CRUISE LINES	\$660,708.00
HOTEL REPS & CHAINS	\$181,234.00
RAILROADS	\$1,850.00
TOUR OPERATORS/WHOLES	\$89,800.00
TRAVEL INSURANCE	\$315.00
YACHT & BOAT CHARTERS	\$19,970.00
(blank)	\$111,939.00
Grand Total	\$2,109,737.00



```

-----
-- Part 2 b)

-- Data Mining
-- Manual SQL Reporting - SUM SUM
-----

2B) Manual Reports

-- SUM SUM of product sales

set null '** Unaccounted sales **'

set line size 80
set pagesize 30

ttitle 'Total Sum of Sales and Running Total'
repheader center 'Total Travel Experts Travel Agency Sales'
btitle '*** Confidential ***'

col product format a35
col total_sales format $9,999,999
col "Prod Running Total" format $9,999,999

select product_name as "Product", sum(base_price) as "Total_Sales",
       to_char(sum(base_price) / sum(sum(base_price)) over() * 100, '9999999.99') as "Total
Percent Sales",
       sum(sum(base_price)) over (order by product_name rows between unbounded
                                     preceding and current row) as "Prod
Running Total"
   from fact_travel group by product_name order by 2 desc;

ttitle off
btitle off
repheader off

```

Sat Mar 18

page 1

Total Sum of Sales and Running Total

Product	Total Travel Experts Travel Agency Sales		
	Total_Sales	Total Perce	Prod Running Total
AIRLINES	\$997,732	47.29	\$1,043,121
CRUISE LINES	\$660,708	31.32	\$1,704,629
HOTEL REPS & CHAINS	\$181,234	8.59	\$1,885,863
** Unaccounted sales **	\$111,939	5.31	\$2,109,737
TOUR OPERATORS/WHOLESALES	\$89,800	4.26	\$1,977,513
AIRLINE CONSOLIDATORS	\$45,389	2.15	\$45,389
YACHT & BOAT CHARTERS	\$19,970	.95	\$1,997,798
RAILROADS	\$1,850	.09	\$1,887,713
CAR RENTALS	\$800	.04	\$1,043,921
TRAVEL INSURANCE	\$315	.01	\$1,977,828

*** Confidential ***

10 rows selected.

Analysis Recommendations

Planning analysis and data mining over the next year provides the opportunity to invest in training and software for the following

Rapid Miner – Supporting features like predictive analysis and machine learning, Rapid Miner can give further insight to the data collected. The machine learning feature will help with future sales predictions with quick report turn around. This will cost \$5000 dollars but comes with a free, annual, or quotebased subscription depending on the additional requested features.

Tableau – Data analysis software will provide the same basic features of Rapid Miner but for far cheaper procurement costs. Tableau procurement cost would be \$500 in comparison to Rapid Miner but also offers subscription or quote based payments as well.

Both software packages allow us to connect the Travel Agency Database for instant reporting. Both also provide data and graph dashboards for immediate viewing.

Marketing Recommendations

The new Travel Experts Data Warehouse can be used going forward to keep track of more than just sales totals. Previously, there was no solution to keep track of the marketing strategies or communications to previous or potential customers. The ability to track communications to the customers has been added with the CUSTOMERCOMM table. Information on the name and last date a customer was contacted, the type of contact (email, cold call, etc) and which employee was reaching out can be tracked to enhance promotions to our most valuable and highest paying customers.

Demographics of the customers can also be tracked by Decision Tree methods. Categorization of clients based on city location and readily available Government of Canada data can show who may typically have more disposable income based on neighborhood. Sales based on quadrants can also be verified with current data mining techniques mentioned earlier.

Further considerations – Data Accuracy

Also shown in the reports pulled is the need to keep the current sales data clean and organized within the database. Seeing over \$100,000 in (blank) product categorization leaves a hole in the data. Such a vast number eclipses the car rentals, travel insurance, and yacht & boat charter sales combined! The need to review this data and cross reference any paperwork is crucial to keep the database updated and accurate.

Travel Experts Travel Agency

Backup and Recovery Plan

February 24th, 2017

Kenny Vigar

kvigar@gmail.com

In the occurrence of a failure regarding the Travel Experts Travel Agency database the following backup methods will be discussed and implemented.

Backup Plan	Page
Background	1
Database Preparation – Design/Gather Files	2
Setup Archive Log Mode	3
Setup Fast Recovery Area	4
Setup Flashback	5
Setup RMAN	6
Recovery Catalog	7
Instance Recovery	8
Calendar Schedule	9
Calendar Legend	10
Disk Strategy	15
Contact Information	16
Appendix 1 – Recovery - 17 – Data File Recovery - 20 – System/Undo Datafile Recovery - 24 – Redo Log Recovery	17
Appendix 2 – Query to gather files for backup	27
Appendix 3 – Linux Backup Shell Script (Syntax Examples)	28

Last revised: March 2017

KV

Background

Travel Experts Travel Agency stores all customer, supplier and sales information within their database and it is vital to the business that this information remain secure and recoverable. The recovery process should be quick, not hampering any sales.

User Errors, Instance Failure, and Media Failure are reasons why you may lose data, or your database. A user for example may accidentally remove data from your records or a computer drive may experience media failure - needing replacement.

Travel Experts business hours are primarily in the day time, which allows for early morning backups without any performance issues occurring on the system. These backups will be an automated script for two reasons - simplicity of scheduling and to avoid human error in duplicating files. The scripts covered in this document will be both user managed MS-DOS based scripts and RMAN scripts. The goal will be to

- Decrease MTTR – Mean Time to Recover – the time it takes to recover the database for use and
- Increase MTBF – Mean Time Between Failure – Maintenance and optimization to keep uptime!

Database Preparation - Protecting Data by Design

To protect against media failure *control files* and online *redo log file* should be mirrored across multiple disks to ensure we have a replacement should a drive stop working. The redo logs have been structured to have 8 groups of 2 members each. This redo log structure will provide no problems keeping recovery information available in the redo logs because of the time it will take to overwrite them. As well, following best practices of using Archive Log Mode (*discussed further below in Archive Log Mode Setup*) to backup these groups before being overwritten will give the option to go to any point in time to recover data to. The redo logs were created during the create database phase and were designed for this purpose of redundancy.

The control file is one of the most critical database components and needs to have many copied available for restoring in the event one gets corrupt or goes missing. It is spanned over 4 disks using the `CONTROL_FILES` parameter in the PFILE. This results in 4 copies of the control file.

Reliable recovery of user data, employee, sales, or customer information. Each logical tablespace in our database will have a corresponding physical file which will be backed up. These tablespaces have been categorized by differing data collections which reduce the size of each tablespace. A smaller tablespace will backup and restore quicker than a larger tablespace. Keeping data separate also keeps only ‘portions’ of data offline if a backup is required in the daytime (or busier periods) so agents can still use other information in the meantime.

Considerations are also made as to which data can be relocated to the read only tablespace. This tablespace is only backed up every 6 months because this data does not change frequently. The more data we can store in read only minimizes the daily backups we need to take on the whole database.

Gather data files – location of all files in Travel Experts Travel Agency

DISK01\CONTROLFILE\CONTROLFILE01.CTL
DISK02\CONTROLFILE\CONTROLFILE02.CTL
DISK03\CONTROLFILE\CONTROLFILE03.CTL
DISK05\CONTROLFILE\CONTROLFILE04.CTL

DISK01\SYSTEM\SYSTEM01.DBF
DISK02\SYSTEM\SYSTEM02.DBF
DISK02\SYSAUX\SYSAUX01.DBF
DISK03\TEMP\TEMP01.TMP
DISK03\DATA\USERDATA01.DBF
DISK04\INDX\INDEX01.DBF
DISK04\READONLY\READONLY01.DBF
DISK05\ADHOC\ADHOC01.DBF
DISK05\UNDOTABLE\UNDOTBS01.DBF

DISK02\archivelog*.ARC
DISK03\archivelog*.ARC

DISK01\REDOLOG\REDO01A.RDO
DISK02\REDOLOG\REDO01B.RDO

DISK01\REDOLOG\REDO02A.RDO
DISK02\REDOLOG\REDO02B.RDO

DISK01\REDOLOG\REDO03A.RDO
DISK02\REDOLOG\REDO03B.RDO

DISK01\REDOLOG\REDO04A.RDO
DISK02\REDOLOG\REDO04B.RDO

DISK01\REDOLOG\REDO05A.RDO
DISK02\REDOLOG\REDO05B.RDO

DISK01\REDOLOG\REDO06A.RDO
DISK02\REDOLOG\REDO06B.RDO

DISK01\REDOLOG\REDO07A.RDO
DISK02\REDOLOG\REDO07B.RDO

Additional files which will be backed up external to the database are

File Location	File Name
%ORACLE_HOME%\NETWORK\ADMIN\	tnsnames.ora
	sqlnet.ora
	listener.ora
%ORACLE_HOME%\database\	PWDtea.ora (Password File)
%ORACLE_HOME%\database\	SPFILETEA.ORA (SPFILE)
DISK03\root	initTEA.ora (PFILE)

* Scripts to gather files are included in Appendix 2

Database Preparation - Archive Log Mode

Oracle can run in two modes regarding recovery, No Archive Log Mode, and Archive Log Mode. It is best practise to run the database in *Archive Log Mode* and is required to support this database's backup plan. When changes are made to the database, any redo (recovery information) will be stored outside of the database in Archive Log files. These log files will allow for recovering past the last consistent backup or recovering data to a specific point in time. RMAN, our primary backup utility also requires Archive Log Mode enabled.

Add the following to the TEA parameter file (inittea.ora) to start Archive Log Mode configuration

Parameter	Description
LOG_ARCHIVE_START=TRUE	Start Archiving Logs in 9i or earlier (enabled by default in 11g)
LOG_ARCHIVE_MAX_PROCESSES=2	Number of processes for ARNc to use (between 1-30)
LOG_ARCHIVE_DEST_n (up to 10 locations)	Location to save the Archived Logs to LOG_ARCHIVE_DEST_1 = "location=C:\TEA\Disk02\archivelog MANDATORY" LOG_ARCHIVE_DEST_2 = "location=C:\TEA\Disk03\archivelog MANDATORY"
LOG_ARCHIVE_MIN_SUCCEED_DEST = 2	Out of the destinations above, how many need to be successful in saving Archived Logs. Providing reliability to our Archive Logs.
LOG_ARCHIVE_FORMAT = %%TEA%%S%r.ARC	Format of the Archive Log file name (T omitted as we're not using RAC, Sequence and Reset number included)

* Verify current LARGE_POOL, MEMORY_TARGET, and MEMORY_MAX_TARGET are suitable for backup processes. The large pool will keep backup process out of the SGA.

- Load SQL PLUS C:\sqlplus /nolog
- Ensure clean database shutdown if running
- Connect as SYS DBA user SQL> conn / as sysdba
- Mount Database SQL> startup mount;
- Start Archive Log Mode SQL> ALTER DATABASE ARCHIVELOG;
- Turn on flashback recover SQL> ALTER DATABASE FLASHBACK ON;
- Verify Archive Log setup is complete SQL> ARCHIVE LOG LIST

```
SQL> SELECT LOG_MODE FROM V$DATABASE;
```

```
SQL> select log_mode from v$database;  
  
LOG_MODE  
-----  
ARCHIVELOG
```

```
SQL> archive log list;  
Database log mode          Archive Mode  
Automatic archival        Enabled  
Archive destination        C:\backup\fra\archivelog  
Oldest online log sequence 74  
Next log sequence to archive 76  
Current log sequence       76
```

*Can force a log archive SQL>ALTER SYSTEM ARCHIVE LOG CURRENT;

Further Archive Log settings can be viewed in the V\$ARCHIVED_LOG, AND V\$ARCHIVE_DEST;

```
SQL> select dest_name, status, target, error, destination from v$archive_dest;  
  
DEST_NAME      STATUS TARGET  ERROR   DESTINATION  
-----  -----  -----  
LOG_ARCHIVE_DEST_1  VALID  PRIMARY      \TEA\Disk02\archivelog  
LOG_ARCHIVE_DEST_2  VALID  PRIMARY      \TEA\Disk03\archivelog  
LOG_ARCHIVE_DEST_3  VALID  PRIMARY      c:\backup\fra\archivelog
```

Database Preparation - Fast Recovery Area

The Fast Recovery Area (FRA) is a more automated option for storing all of the files required to recover the database. The Fast recovery area allows files to be backed up to and automatically removed if they become obsolete. The FRA should be large enough to hold permanent files (multiplexed control file and online redo logs). The archive logs will also be set to backup to the FRA. Flashback Logs (for database flashback) will be stored here as well.

```
LOG_ARCHIVE_DEST_3 = "location=C:\backup\fra\archivelog MANDATORY"
```

RMAN will also automatically backup to the Flash Recovery Area as well, unless we specify a different location for the backups to go to. The RMAN backups in the FRA will be automatically kept or deleted by our retention configuration (discussed next).

FRA Parameters

Parameter	Description
DB_RECOVERY_FILE_DEST='c:\backup\fra';	Location of the central backup location (FRA) – which is placed on a separate disk than the database data files
DB_RECOVERY_DEST_SIZE=2G;	Size of fast recovery area. Should be double your database size, minimum.

We can monitor the Fast Recovery Area with the V\$RECOVERY_FILE_DEST view or can use Enterprise Manager to view FRA current settings by clicking Availability > Recovery Settings.

Database Preparation - Flash Back Setup

Flash Back allows us to recover from any logical corruption. If a table is modified with an insert, update or delete, recovery to the previous version of this data.

To configure the database for Flashback set the following parameters in the PFILE.

Parameter	Description
UNDO_MANAGEMENT='auto'	Auto management of undo segments
UNDO_TABLESPACE='undotbs1'	Tablespace specified for undo management
UNDO_RETENTION=500	Hold flashback undo for 500 seconds.

To view the data as it existed before the error and before recovering it use a Flashback Query command.

```
SQL> SELECT customer_ref, first_name, last_name  
AS OF timestamp TO_TIMESTAMP ('2017-02-23', 'YYYY-MM-DD HH24:MI:SS')  
WHERE customer_ref = 200;
```

Another option is to run a Flash Back Version Query to see all changes of the data as it existed between two points in time.

If data in one tablespace is extremely important we can set a Guarantee Retention policy to make sure the data is available as long as we need and is not lost from our Undo tablespace.

```
SQL> ALTER TABLESPACE companydata RETENTION GUARANTEE;
```

Database Preparation - RMAN Configuration

As mentioned previously, RMAN will be our primary utility for backing up the database. To start RMAN

```
C:\Windows\System32>rman target / catalog rcatowner/oracle@teatest

Recovery Manager: Release 11.2.0.1.0 - Production on Mon Feb 20 14:01:29 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

connected to target database: TEA (DBID=2506821537)
connected to recovery catalog database

RMAN>
```

If RMAN will be backing up to a tape device, we will need to setup and install the Media Management Layer software. This is specific to the vendor and their documentation will need to be followed. RMAN parameters regarding SBT (tape backup) are configured below for reference but this backup plan will primarily use disk backups.

Parameters to configure in RMAN are

RMAN Config Command	Description
CONFIGURE CONTROLFILE AUTOBACKUP ON;	Automatically backs up Control File on every backup
CONFIGURE DEVICE TYPE DISK PARALLELISM 3;	Set Disk Backup parallel processes to 3
CONFIGURE DEVICE TYPE SBT PARALLELISM 3;	Set Tape Backup parallel processes to 3
CONFIGURE DEVICE TYPE DISK BACKUP TYPE TO COMPRESSED BACKUPSET;	Set Disk backups to be backed up as a compressed backup set
CONFIGURE DEVICE TYPE SBT BACKUP TYPE TO COMPRESSED BACKUPSET;	Set Tape backups to be backed up as a compressed backup set
CONFIGURE SNAPSHOT CONTROLFILE NAME TO 'C:\backup\rman\SNCFTEA.ORA';	Location of autobackup of Snapshot Control File

Retention Policy Settings

RMAN Configuration also is used to control the automatic deletion of backups in the FRA. These deletions are controlled by retention policies. We will be using a *Recovery Window* for Travel Agents to ensure we can recover any time in the last 7 days.

```
RMAN> CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 7 DAYS;
old RMAN configuration parameters:
CONFIGURE RETENTION POLICY TO REDUNDANCY 2;
new RMAN configuration parameters:
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 7 DAYS;
```

```
new RMAN configuration parameters are successfully stored
starting full resync of recovery catalog
full resync complete
```

Encryption Settings for RMAN

Also important in configuring RMAN is to have the global encryption setting on. Now any backups occurring will be encrypted with AES128 encrypted algorithm.

```
RMAN> CONFIGURE ENCRYPTION FOR DATABASE on;
      new RMAN configuration parameters:
      CONFIGURE ENCRYPTION FOR DATABASE ON;
      new RMAN configuration parameters are successfully stored
      starting full resync of recovery catalog
      full resync complete

RMAN> show encryption algorithm;
      RMAN configuration parameters for database with db_unique_name TEA are:
      CONFIGURE ENCRYPTION ALGORITHM 'AES128'; # default
RMAN> show ENCRYPTION FOR DATABASE;
      RMAN configuration parameters for database with db_unique_name TEA are:
      CONFIGURE ENCRYPTION FOR DATABASE ON;
```

Recovery Catalog Setup

Metadata on our backups from RMAN are stored in the control file for our database. If anything happens to our control file we might lose this data. We can backup our backup data in a separate schema – The Recovery Catalog – in a separate non production database. Our Travel Agent recovery catalog will be stored in our TEATEST database.

Setup is detailed below,

-Verify Oracle Sid (c:\set oracle_sid) is correct (TEATEST) for the database RCAT schema and tablespace will be created in.

```
SQL> CREATE TABLESPACE rcat DATAFILE 'C:\app\Administrator\oradata\orant11g\rcat.dbf' SIZE 15M;

SQL> CREATE USER rcatowner
      IDENTIFIED BY oracle
      TEMPORARY TABLESPACE temp
      DEFAULT TABLESPACE rcat
      QUOTA UNLIMITED ON rcat;

SQL> GRANT recovery_catalog_owner TO rcatowner;
```

Load RMAN and create and sync catalog

```
c:\>rman catalog rcatowner/oracle@teatest

RMAN> create catalog;
RMAN> register database;
      database registered in recovery catalog
      starting full resync of recovery catalog
      full resync complete
RMAN> resync catalog;
```

There will be a second recovery catalog created in the TEA database to back up the TEATEST recovery catalog. The steps for creation will be the same as above. The TEATEST database is also backed up with the same scheduling discussed later in this document. The recovery catalog will be protected from loss this way.

Instance Recovery

When the database instance hangs – shutting down and restarting will allow SMON processes to load redo data automatically. SMON applies all data in redo logs for data availability the user and rolls back uncommitted data once it's available.

Instance recovery will be set by adding the parameter FAST_START_MTTR_TARGET to our PFILE.

PFILE Parameter	Description
FAST_START_MTTR_TARGET=600	Tunes Instance Recovery to 5 minutes
RECOVERY_PARALLELISM=4	Roll Forward Tuning
FAST_START_PARALLEL_ROLLBACK=HIGH	Roll Back Tuning Additional SMON slaves

The FAST_START_MTTR_TARGET sets a checkpoint for the database writer process (DBWr) to write redo logs to archive logs to ensure the recovery will take no longer than 5 minutes. These checkpoints signal when to start instance recover for the database on startup. As well as setting this parameter – instance recovery can be further optimized by setting smaller online redo log file sizes.

Because the parameter FAST_START_MTTR_TARGET is used there is no need to use parameters LOG_CHECKPOINT_TIMEOUT or LOG_CHECKPOINT_INTERVAL.

Monitoring the Instance Recovery can be performed by querying the V\$INSTANCE_RECOVERY view.

* It is highly recommended you back up the control file when instance recovery has occurred. It is also recommended that the Alert Log be checked after any Instance Recovery occurs. Trends in this behavior will need to be analyzed.

MAY						
M BUSINESS HOURS 9AM-5PM	T BUSINESS HOURS 9AM-5PM	W BUSINESS HOURS 9AM-5PM	T BUSINESS HOURS 9AM-9PM	F BUSINESS HOURS 9AM-5PM	S BUSINESS HOURS 9AM-5PM	S BUSINESS HOURS CLOSED
The ALERT LOG stored in disk03\diag\ diagnostics should be viewed daily	1 6:00am INCREM1	2 6:00am INCREM1C	3 6:00am INCREM1	4 6:00am INCREM1	5 6:00am INCREM1	6 8:00am FullDBCOPY
	6:30am CTL_NET_SP	6:30am USR_MG_DBF	6:30am DP_EXP	7:00pm PRAC	6:30am DBF	9:00am FullDBBACKUPSET
						10:00am INCREMO
7 6:00am INCREM1	8 6:00am INCREM1	9 6:00am INCREM1C	10 6:00am INCREM1	11 6:00am INCREM1	12 6:00am INCREM1	13 9:00am FullDBBACKUPSET
11:30am MGR REVIEW	6:30am CTL_NET_SP	6:30am USR_MG_DBF	6:30am DP_EXP	7:00pm PRAC	6:30am DBF	10:00am INCREMO
14 6:00am INCREM1	15 6:00am INCREM1	16 6:00am INCREM1C	17 6:00am INCREM1	18 6:00am INCREM1	19 6:00am INCREM1	20 9:00am FullDBBACKUPSET
	6:30am CTL_NET_SP	6:30am USR_MG_DBF	6:30am DP_EXP	7:00pm PRAC	6:30am DBF	10:00am INCREMO
21 6:00am INCREM1	22 6:00am INCREM1	23 6:00am INCREM1C	24 6:00am INCREM1	25 6:00am INCREM1	26 6:00am INCREM1	27 9:00am FullDBBACKUPSET
	6:30am CTL_NET_SP	6:30am USR_MG_DBF	6:30am DP_EXP	7:00am PRAC	6:30am DBF	10:00am INCREMO
28 6:00am INCREM1	29 6:00am INCREM1	30 6:00am INCREM1C	31 6:00am INCREM1			11:00am ARCHLOG
	6:30am CTL_NET_SP	6:30am USR_MG_DBF	6:30am DP_EXP			

*Legend

FullDBCOPY - 8:00am FIRST SUNDAY

First Sunday of *each month* at 8am is a full copy of the database for the previous month before it.

RMAN> BACKUP AS COPY DATABASE PLUS ARCHIVELOG;

RMAN> LIST COPY; -- to verify backupset dates/files.

The database will be shut down completely for a consistent backup.

FullDBBackupSet - 9:00am WEEKLY SUNDAY

Every Sunday at 9am full backupset of the database will be stored for a weekly backup.

```
RMAN> BACKUP AS BACKUPSET DATABASE FORMAT 'c:\backup\rman\dbbackup_%f_%c',
'c:\backup\rman2\dbbackup_%f_%c';
```

RMAN> LIST BACKUP SET; -- to verify backupset dates/files.

Because we have the FRA setup to start a backup type the syntax below. You do not need to type a location of the backup sets with the FOMAT option above. To save to as it was dictated by the DB_RECOVERY_FILE_DEST= parameter use command.

```
RMAN> backup as backup set database plus archivelog;
```

Backup sets will be monitored with v\$backup_set and v\$backup_datafile;

```
run {
  SQL 'alter system archive log current';
  ALLOCATE CHANNEL chan1 TYPE disk FORMAT 'c:\backup\rman\database_files_1_2_3_%u';
  ALLOCATE CHANNEL chan2 TYPE disk FORMAT 'c:\backup\rman\database_files_4_5_6_%u';
  ALLOCATE CHANNEL chan3 TYPE disk FORMAT 'c:\backup\rman\database_files_7_8_9_%u';
  ALLOCATE CHANNEL chan4 TYPE disk FORMAT 'c:\backup\rman\database_files_10_11_%u';
  BACKUP AS BACKUPSET INCREMENTAL LEVEL = 1
    (datafile 1,2,3 channel chan1)
    (datafile 4,5,6 channel chan2)
    (datafile 7,8,9 channel chan3)
    (datafile 10,11 channel chan4);
  SET BACKUP COPIES 2;
  BACKUP AS BACKUPSET DATABASE FORMAT 'c:\backup\rman\dbbackup_%f_%c',
'c:\backup\rman2\dbbackup_%f_%c';
  SQL 'alter system archive log current';
}
```

INCREM0 - 10:00am WEEKLY SUNDAY

Sunday will also be the start of our level 0 incremental backups. The RMAN script below parallels these backups to run quickly. This particular script is created for a device type disk backup (as opposed to a sbt tape device type). A control file is backed up here as well.

```
RMAN>
run { set CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE DISK TO 'c:\backup\rman\ctl_%F';
ALLOCATE CHANNEL chan1 TYPE disk FORMAT 'c:\backup\rman\increm_files_1_2_3_%u';
ALLOCATE CHANNEL chan2 TYPE disk FORMAT 'c:\backup\rman\increm_files_4_5_6_%u';
ALLOCATE CHANNEL chan3 TYPE disk FORMAT 'c:\backup\rman\increm_files_7_8_9_%u';
ALLOCATE CHANNEL chan4 TYPE disk FORMAT 'c:\backup\rman\increm_files_10_11_%u';
BACKUP AS BACKUPSET INCREMENTAL LEVEL = 0
  (datafile 1,2,3 channel chan1)
  (datafile 4,5,6 channel chan2)
  (datafile 7,8,9 channel chan3)
  (datafile 10,11 channel chan4);}
```

INCREM1 - 6:00am MONDAY TUESDAY THURSDAY FRIDAY SATURDAY

Monday, Tuesday, Thursday, Friday and Saturday at 6am run an incremental backup on the same datafiles which were backed up in INCREM0. The difference between INCREM1 and INCREM0 is BACKUP AS BACKUPSET INCREMENTAL LEVEL = 1. A control file is also backed up here.

INCREM1C - 6:00am WEDNESDAY

Is performed on Wednesday daily at 6am. INCREM1 cumulates the backups before it allowing for quicker restore times if there is a need later in the week. It also runs the same script, but with the difference BACKUP AS BACKUPSET INCREMENTAL LEVEL = 1C. Again, a control file is also backed up at this backup level.

DBF – 6:30am WEEKLY SATURDAY

RMAN can backup our datafiles individually as well, aside from the full copy/dataset and incremental backups. These are performed Saturday at 6:30am. To find file locations in RMAN use

```
RMAN> report schema;

Report of database schema for database with db_unique_name TEA

List of Permanent Datafiles
=====
File Size(MB) Tablespace  RB segs   Datafile Name
-----  -----  -----  -----
```

1	200	SYSTEM	YES	C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF
2	200	SYSTEM	YES	C:\TEA\DISK04\SYSTEM\SYSTEM01.DBF
3	1100	SYSAUX	NO	C:\TEA\DISK02\SYSAUX\SYSAUX01.DBF
4	150	UNDOTBS	YES	C:\TEA\DISK05\UNDOTABLE\UNDOTBS01.DBF
5	50	USERDATA	NO	C:\TEA\DISK03\DATA\USERDATA01.DBF
6	50	SUPPLIERDATA	NO	C:\TEA\DISK03\DATA\SUPPLIERDATA01.DBF
7	50	ADHOC	NO	C:\TEA\DISK05\ADHOC\ADHOC01.DBF
8	50	INDX	NO	C:\TEA\DISK04\INDX\INDEX01.DBF
9	50	READONLY	NO	C:\TEA\DISK04\READONLY\READONLY01.DBF
10	50	COMPANYDATA	NO	C:\TEA\DISK03\DATA\COMPANYDATA01.DBF
11	700	XDB	NO	C:\TEA\DISK02\XDB\XDB01.DBF

List of Temporary Files

=====

File Size(MB)	Tablespace	Maxsize(MB)	Tempfile Name
---------------	------------	-------------	---------------

1	150	TEMP	200	C:\TEA\DISK03\TEMP\TEMP01.TMP
---	-----	------	-----	-------------------------------

To backup datafiles use syntax below (or script similar to page 12's incremental data file script for all datafiles).

```
RMAN> backup datafile 3 format 'c:\backup\database_files_3_%u';

Starting backup at 20-FEB-17
using channel ORA_DISK_1
using channel ORA_DISK_2
using channel ORA_DISK_3
channel ORA_DISK_1: starting compressed full datafile backup set
channel ORA_DISK_1: specifying datafile(s) in backup set
input datafile file number=00003 name=C:\TEA\DISK02\SYSAUX\SYSAUX01.DBF
channel ORA_DISK_1: starting piece 1 at 20-FEB-17
channel ORA_DISK_1: finished piece 1 at 20-FEB-17
piece handle=C:\BACKUP\DATABASE_FILES_3_AERT3623 tag=TAG20170220T220803 comment=NONE
channel ORA_DISK_1: backup set complete, elapsed time: 00:03:12
Finished backup at 20-FEB-17
```

```
Starting Control File and SPFILE Autobackup at 20-FEB-17
piece handle=C:\BACKUP\RMAN\CTL_C-2506821537-20170220-08 comment=NONE
Finished Control File and SPFILE Autobackup at 20-FEB-17
```

USR_MG_DBF - 6:30am WEDNESDAY

User Managed backup scripts run Wednesdays at 6:30am. A script runs to place a tablespace in Begin Backup Mode. A copy of the data file runs, then the script places the tablespace out of Backup Mode. Each datafile is then

verified with the DBVERIFY utility.

A sample of the CompanyData01 tablespace is below. Each tablespace is backed up the same way with a separate .sql script which is called from the initial batch file.

```
C:\ sqlplus /nolog @C:\TEA\buscript\9_CompanyData.sql
SQL>spool c:\backup\backup_log.txt append
SQL>SET ECHO ON

SQL>conn / as sysdba

SQL>Alter TABLESPACE companydata BEGIN BACKUP;

SQL>disconnect
SQL>exit
SQL>spool off

C:\copy C:\TEA\DISK03\DATA\COMPANYDATA01.DBF c:\backup\_currbackup\COMPANYDATA01.DBF

SQL>Spool c:\backup\backup_log.txt append

SQL>SET ECHO ON
SQL>conn / as sysdba

SQL>Alter TABLESPACE companydata END BACKUP;

SQL>disconnect
SQL>exit
SQL>spool off
```

DP_EXT 6:30am THURSDAY

A logical export of the database will occur weekly at 6:30am Wednesdays. The datapump export provides a dump file which can be used to recover the DDL used to create the tables and can export indexes if needed for recovery.

```
C:\expdp teaadmin/teaadmin123 directory=backup_dir dumpfile=tea.dmp logfile=tea_tempdata.log
```

CTL_NET_SP - Further File Backups

In addition to the automatic control file backups above a user managed backup script will run every Tuesday at 630am for explicit Control File backup to script (trace) file and BKP file. The database will be shut down for this backup.

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO 'c:\backup\_currbackup\control.bkp';
Database altered.
```

```
SQL> ALTER DATABASE BACKUP CONTROLFILE TO TRACE AS 'c:\backup\_currbackup\controlfile.trc';
Database altered.
```

This time is also given to perform O/S copies of the network files (TNSNAMES.ORA, SQLNET.ORA, LISTENER.ORA(s) stored in

```
Location - C:\app\Administrator\product\11.2.0\dbhome_1\NETWORK\ADMIN
```

```
OS COPY - C:\>copy C:\app\Administrator\product\11.2.0\dbhome_1\NETWORK\ADMIN\*.ora
C:\backup\_currbackup\
```

The SPFile is also backed up

```
SQL> CREATE SPFILE = 'c:\backup\spfiletea.ora' FROM PFILE = 'c:\tea\inittea.ora';
File created.
```

```
SQL> CREATE SPFILE = 'c:\backup\spfileteamem.ora' from memory;
File created.
```

ARCHLOG – EVERY SECOND MONDAY 2:00pm

The archive logs are backed up with the FULLDBCOPY. Archive Log locations will be cleared once a month on the last Sunday after the full backup copy.

```
\TEA\Disk02\archivelog
\TEA\Disk03\archivelog
```

Archive Log locations will be checked and cleared once a month. We will be checking the LOG_ARCHIVE_DEST_ folders to make sure the files are not growing too large and to clear archive logs no longer required.

To check logs and delete logs, use RMAN commands,

```
RMAN> LIST ARCHIVELOG ALL;
RMAN> DELETE ARCHIVELOG ALL
```

Yearly Backup

A monthly backup is kept (described above) but not shown on the calendar is the yearly backup. The first Sunday of the year will have a backup run over the previous year's data. This backup will be kept for 7 years for any auditing reasons.

```
RMAN> BACKUP DATABASE PLUS ARCHIVELOG KEEP UNTIL TIME 'sysdate+2555'
FORMAT 'c:\backup\yearly_db_backup_%u' TAG TEA_YEARLY;
```

MGR REVIEW - 11:30am MONDAY

This is our monthly scheduled meetings with management to discuss the current backup plan and to advise of any changes to the plan.

PRAC – 7am FRIDAY

This time is scheduled for no users interacting with the database to allow the DBA to test current recovery strategies. The practice occurs the Friday before the manager review. Any issues with the backup process or changes can be determined here and discussed the following week with management.

Tape and Offsite Storage

Backups will be stored to a solid state hard drive housed in an external enclosure. The database has been configured to allow tape storage, but they will not be used at this time.

Monthly Backups

- 24 disks will be needed for full month backups. These are the **FullDBCopys** that occur on the first Sunday of each month. A backup will be copied to two separate disks. One will be kept onsite in a locked, fireproof safe. The other will be stored offsite at Iron Mountian (5811 26 St SE Calgary AB)

Weekly Backups

- A weeks-worth of backup data will be kept on hand in office, and off site at a storage facility (Iron Mountain 5811 26 St SE, Calgary) these are **FullDBBACKUPSET** and **INCREM0** backups. A total of 10 disks will be needed to keep 5 weeks of data (duplicated).
- Copies of the control files, network files, and SPFILE will be backed to the same disk the following Tuesday before being shipped off site.

Daily Backups

- These incremental backups (INCREM1 and INCREM1C) are stored on site in a fireproof locked safe. There will be 7 tapes for the daily backups. They will be overwritten each week as a week's worth of data is kept on and off site in our weekly backups.

Yearly Backup

- A yearly backup will be taken on the first Sunday (as mentioned) of the year at 6am and duplicated to 2 disks. These will be stored at Iron Mountain and the manager's residence. No full year backup will be stored on site.

Contact Information Should Issues Arise (Who can you depend on?)

Person/Company	Phone Number	Escalation Procedure
Kenny Vigar	403.870.9208	DBA On Call Contact for any questions regarding documentation, database backup issues or recovery issues not covered in this plan
Greg Smith	1.650.506-7000	Oracle Technical Support Contact with any escalations the DBA On Call cannot resolve
Iron Mountain	403.531.2000	Iron Mountain Tape Storage Contact for any requests to have tape backups shipped to head office
HP Server Support	1.800.334.5144	HP Technical Support Contact for any issues with server hardware (disk failures or other hardware)

* Note that management/owner are not on the escalation list. They should be notified of any issues but were not included as they are not technical resources. Keep management informed at each escalation phase.

Appendix 1 - Recovery

In the event of data loss or the need to perform full disaster recovery this documentation will be used. RMAN again will be our primary recovery tool. Each file listed in Travel Experts database can be recovered in the following steps.

Non Critical Datafile Restore and Recovery

A non-critical datafile may go missing or go corrupt. Below is an excerpt from the alert_tea.log file.

```
ALTER DATABASE OPEN
Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_dbw0_2476.trc:
ORA-01157: cannot identify/lock data file 10 - see DBWR trace file
ORA-01110: data file 10: 'C:\TEA\DISK03\DATA\COMPANYDATA01.DBF'
ORA-27046: file size is not a multiple of logical block size
OSD-04012: file size mismatch (OS 52445204)
Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_ora_9124.trc:
ORA-01157: cannot identify/lock data file 10 - see DBWR trace file
ORA-01110: data file 10: 'C:\TEA\DISK03\DATA\COMPANYDATA01.DBF'
ORA-1157 signalled during: ALTER DATABASE OPEN...
Mon Feb 20 23:50:43 2017
Checker run found 1 new persistent data failures
```

Query V\$RECOVER_FILE to see file# with problem and confirm alert log.

```
SQL> SELECT rf.FILE#, df.name, rf.online_status, rf.error
      FROM v$recover_file rf join v$datafile df ON rf.file#=df.file#;

  FILE#    NAME          ONLINE_STATUS    ERROR
-----  -----
    10    C:\TEA\DISK03\DATA\COMPANYDATA01.DBF    ONLINE FILE    NOT FOUND
```

Confirm in RMAN

```
C:\Windows\System32>rman target / catalog=rcatowner/oracle@teatest

Recovery Manager: Release 11.2.0.1.0 - Production on Wed Feb 22 14:15:13 2017
Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
connected to target database: TEA (DBID=2506821537)
connected to recovery catalog database

RMAN> list failure;
```

List of Database Failures

Failure ID	Priority	Status	Time Detected	Summary
------------	----------	--------	---------------	---------

11282	HIGH	OPEN	20-FEB-17	One or more non-system datafiles are corrupt
-------	------	------	-----------	--

Restore Datafile 10

```
RMAN> restore datafile 10;
```

```
Starting restore at 21-FEB-17
```

```
allocated channel: ORA_DISK_1
```

```
channel ORA_DISK_1: SID=10 device type=DISK
```

```
allocated channel: ORA_DISK_2
```

```
channel ORA_DISK_2: SID=97 device type=DISK
```

```
allocated channel: ORA_DISK_3
```

```
channel ORA_DISK_3: SID=11 device type=DISK
```

```
channel ORA_DISK_1: starting datafile backup set restore
```

```
...
```

```
...
```

```
channel ORA_DISK_1: specifying datafile(s) to restore from backup set
```

```
channel ORA_DISK_1: restoring datafile 00010 to C:\TEA\DISK03\DATA\COMPANYDATA01.DBF
```

```
channel ORA_DISK_1: reading from backup piece C:\BACKUP\RMAN\INCREM_FILES_10_11_B7RT372R
```

```
channel ORA_DISK_1: piece handle=C:\BACKUP\RMAN\INCREM_FILES_10_11_B7RT372R
```

```
tag=TAG20170220T222524
```

```
channel ORA_DISK_1: restored backup piece 1
```

```
channel ORA_DISK_1: restore complete, elapsed time: 00:00:03
```

```
Finished restore at 21-FEB-17
```

Recover Datafile 10

```
RMAN> recover datafile 10;
```

```
Starting recover at 21-FEB-17
```

```
using channel ORA_DISK_1
```

```
using channel ORA_DISK_2
```

```
using channel ORA_DISK_3
```

```
starting media recovery
```

```
media recovery complete, elapsed time: 00:00:01
```

```
Finished recover at 21-FEB-17
```

```
RMAN> Validate datafile 10;
```

```
Starting validate at 22-FEB-17
```

```
using channel ORA_DISK_1
```

```
using channel ORA_DISK_2
```

```
using channel ORA_DISK_3
```

```
channel ORA_DISK_1: starting validation of datafile
```

```
channel ORA_DISK_1: specifying datafile(s) for validation
```

```
input datafile file number=00010 name=C:\TEA\DISK03\DATA\COMPANYDATA01.DBF
```

```
channel ORA_DISK_1: validation complete, elapsed time: 00:00:01
```

```
List of Datafiles
```

```
=====
```

```
File Status Marked Corrupt Empty Blocks Blocks Examined High SCN
```

```
-----
```

10	OK	0	13	3201	11659606
----	----	---	----	------	----------

```
File Name: C:\TEA\DISK03\DATA\COMPANYDATA01.DBF
```

```
Block Type Blocks Failing Blocks Processed
```

```
-----
```

Data	0	334
------	---	-----

Index	0	0
-------	---	---

Other	0	2853
-------	---	------

```
Finished validate at 22-FEB-17
```

Critical Datafile Restore and Recovery with RMAN

```
SQL> ALTER DATABASE OPEN

Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_dbw0_5292.trc:
ORA-01157: cannot identify/lock data file 1 - see DBWR trace file
ORA-01110: data file 1: 'C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF'

ORA-27041: unable to open file
OSD-04002: unable to open file
O/S-Error: (OS 2) The system cannot find the file specified.

Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_ora_3128.trc:
ORA-01157: cannot identify/lock data file 1 - see DBWR trace file
ORA-01110: data file 1: 'C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF'
ORA-1157 signalled during: ALTER DATABASE OPEN...
Tue Feb 21 00:42:30 2017
Checker run found 1 new persistent data failures
```

Query V\$RECOVER_FILE to see file# with problem and confirm alert log. Verify Mount to replace Critical File.

```
SQL> SELECT rf.FILE#, df.name, rf.online_status, rf.error
   FROM v$recover_file rf join v$datafile df ON rf.file#=df.file#;

  FILE#      NAME          ONLINE_ERROR
  ----- -----
    1  C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF    ONLINE FILE NOT FOUND

SQL> select status from v$instance;

  STATUS
  -----
MOUNTED
```

```
RMAN> restore datafile 1;

Starting restore at 21-FEB-17
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=10 device type=DISK
allocated channel: ORA_DISK_2
channel ORA_DISK_2: SID=95 device type=DISK
allocated channel: ORA_DISK_3
channel ORA_DISK_3: SID=11 device type=DISK
```

```
channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: specifying datafile(s) to restore from backup set
channel ORA_DISK_1: restoring datafile 00001 to C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF
channel ORA_DISK_1: reading from backup piece C:\BACKUP\RMAN\INCREM_FILES_1_2_3_B4RT372K
channel ORA_DISK_1: piece handle=C:\BACKUP\RMAN\INCREM_FILES_1_2_3_B4RT372K
tag=TAG20170220T222524
channel ORA_DISK_1: restored backup piece 1
channel ORA_DISK_1: restore complete, elapsed time: 00:00:07
Finished restore at 21-FEB-17
```

```
RMAN> recover datafile 1;

Starting recover at 22-FEB-17
using channel ORA_DISK_1
using channel ORA_DISK_2
using channel ORA_DISK_3
RMAN> Validate datafile 1;

Starting validate at 22-FEB-17
using channel ORA_DISK_1
using channel ORA_DISK_2
using channel ORA_DISK_3
channel ORA_DISK_1: starting validation of datafile
channel ORA_DISK_1: specifying datafile(s) for validation
including current SPFILE in backup set
including current control file for validation
input datafile file number=00001 name=C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF
channel ORA_DISK_1: validation complete, elapsed time: 00:00:03
List of Control File and SPFILE
=====
File Type Status Blocks Failing Blocks Examined
-----
SPFILE OK 0 2
Control File OK 0 662
List of Datafiles
=====
File Status Marked Corrupt Empty Blocks Blocks Examined High SCN
-----
1 OK 0 3996 12800 11807386
File Name: C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF
```

Block Type	Blocks	Failing Blocks	Processed
Data	0	6548	
Index	0	1733	
Other	0	523	

Data	0	6548
Index	0	1733
Other	0	523

Finished validate at 22-FEB-17

RMAN>
starting media recovery
media recovery complete, elapsed time: 00:00:01

Finished recover at 21-FEB-17

RMAN> Validate datafile 1;

Starting validate at 22-FEB-17
using channel ORA_DISK_1
using channel ORA_DISK_2
using channel ORA_DISK_3
channel ORA_DISK_1: starting validation of datafile
channel ORA_DISK_1: specifying datafile(s) for validation
including current SPFILE in backup set
including current control file for validation
input datafile file number=00001 name=C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF
channel ORA_DISK_1: validation complete, elapsed time: 00:00:03

List of Control File and SPFILE

=====

File Type	Status	Blocks	Failing Blocks	Examined
SPFILE	OK	0	2	
Control File	OK	0	662	

SPFILE	OK	0	2
Control File	OK	0	662
List of Datafiles			
=====			

File Status	Marked	Corrupt	Empty	Blocks	Blocks Examined	High SCN
1	OK	0	3996	12800	11807386	

File Name: C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF

Block Type Blocks Failing Blocks Processed

Data	0	6548
Index	0	1733

Other 0 523

Finished validate at 22-FEB-17

RMAN>

Recovery – Redo Log Group

Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_lgwr_8748.trc:
ORA-00313: open failed for members of log group 1 of thread 1
ORA-00312: online log 1 thread 1: 'C:\TEA\DISK02\REDOLOG\REDO01B.RDO'
ORA-27041: unable to open file
OSD-04002: unable to open file
O/S-Error: (OS 2) The system cannot find the file specified.
Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_lgwr_8748.trc:
ORA-00321: log 1 of thread 1, cannot update log file header
ORA-00312: online log 1 thread 1: 'C:\TEA\DISK02\REDOLOG\REDO01B.RDO'
Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_lgwr_8748.trc:
ORA-00313: open failed for members of log group 2 of thread 1
ORA-00312: online log 2 thread 1: 'C:\TEA\DISK02\REDOLOG\REDO02B.RDO'
ORA-27041: unable to open file
OSD-04002: unable to open file
O/S-Error: (OS 2) The system cannot find the file specified.
Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_lgwr_8748.trc:
ORA-00321: log 2 of thread 1, cannot update log file header
ORA-00312: online log 2 thread 1: 'C:\TEA\DISK02\REDOLOG\REDO02B.RDO'
Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_lgwr_8748.trc:
ORA-00313: open failed for members of log group 3 of thread 1
ORA-00312: online log 3 thread 1: 'C:\TEA\DISK02\REDOLOG\REDO03B.RDO'
ORA-27041: unable to open file
OSD-04002: unable to open file
O/S-Error: (OS 2) The system cannot find the file specified.
Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_lgwr_8748.trc:
ORA-00321: log 3 of thread 1, cannot update log file header
ORA-00312: online log 3 thread 1: 'C:\TEA\DISK02\REDOLOG\REDO03B.RDO'
Errors in file c:\tea\disk03\diagnostic\diag\rdbms\tea\tea\trace\tea_lgwr_8748.trc:
ORA-00313: open failed for members of log group 4 of thread 1
ORA-00312: online log 4 thread 1: 'C:\TEA\DISK02\REDOLOG\REDO04B.RDO'
ORA-27041: unable to open file
OSD-04002: unable to open file
O/S-Error: (OS 2) The system cannot find the file specified.

Confirm in RMAN

```
C:\Windows\System32>rman target / catalog=rcatowner/oracle@teatest

Recovery Manager: Release 11.2.0.1.0 - Production on Wed Feb 22 14:26:11 2017
Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
connected to target database: TEA (DBID=2506821537)
connected to recovery catalog database

RMAN> list failure;

List of Database Failures
=====
Failure ID Priority Status Time Detected Summary
-----
11771 HIGH OPEN 22-FEB-17 Redo log file C:\TEA\DISK02\REDOLOG\REDO04B.RDO is missing
11768 HIGH OPEN 22-FEB-17 Redo log file C:\TEA\DISK02\REDOLOG\REDO03B.RDO is missing
11765 HIGH OPEN 22-FEB-17 Redo log file C:\TEA\DISK02\REDOLOG\REDO02B.RDO is missing
11762 HIGH OPEN 22-FEB-17 Redo log file C:\TEA\DISK02\REDOLOG\REDO01B.RDO is missing

RMAN>
```

Confirm in SQLPLUS

```
SQL> SELECT * FROM v$logfile;

GROUP# STATUS TYPE MEMBER
-----
1 ONLINE C:\TEA\DISK01\REDOLOG\REDO01A.RDO
1 INVALID C:\TEA\DISK02\REDOLOG\REDO01B.RDO
4 ONLINE C:\TEA\DISK01\REDOLOG\REDO04A.RDO
4 INVALID C:\TEA\DISK02\REDOLOG\REDO04B.RDO
2 ONLINE C:\TEA\DISK01\REDOLOG\REDO02A.RDO
2 INVALID C:\TEA\DISK02\REDOLOG\REDO02B.RDO
3 ONLINE C:\TEA\DISK01\REDOLOG\REDO03A.RDO
3 INVALID C:\TEA\DISK02\REDOLOG\REDO03B.RDO
```

Verify Active Logs

```
SQL> SELECT * FROM v$log;

GROUP# THREAD# SEQUENCE# BYTES BLOCKSIZE MEMBERS ARC STATUS
-----
1 1 133 31457280 512 2 YES INACTIVE
2 1 135 31457280 512 2 NO CURRENT
3 1 134 31457280 512 2 YES INACTIVE
4 1 132 10485760 512 2 YES INACTIVE
```

Clear Log File groups which are not active

```
SQL> alter database clear logfile group 1;
```

Database altered.

```
SQL> alter database clear logfile group 2;
```

```
alter database clear logfile group 2
```

```
SQL> alter database clear logfile group 3;
```

Database altered.

```
SQL> alter database clear logfile group 4;
```

Database altered.

Switch log file and clear last Online Redo Log group

```
SQL> alter system switch logfile;
```

System altered.

Verify log file status again – Status is clear

```
SQL> SELECT * FROM v$logfile;
```

GROUP#	STATUS	TYPE	MEMBER
1	ONLINE		C:\TEA\DISK01\REDOLOG\REDO01A.RDO
1	ONLINE		C:\TEA\DISK02\REDOLOG\REDO01B.RDO
4	ONLINE		C:\TEA\DISK01\REDOLOG\REDO04A.RDO
4	ONLINE		C:\TEA\DISK02\REDOLOG\REDO04B.RDO
2	ONLINE		C:\TEA\DISK01\REDOLOG\REDO02A.RDO
2	ONLINE		C:\TEA\DISK02\REDOLOG\REDO02B.RDO
3	ONLINE		C:\TEA\DISK01\REDOLOG\REDO03A.RDO
3	ONLINE		C:\TEA\DISK02\REDOLOG\REDO03B.RDO

Appendix 2

Identify files to back up

Datafiles to Collect				
<pre>SQL> SELECT df.file# "Data_File_Num", ts.name "Table Space Name", df.name Location, df.bytes Bytes, ((df.bytes/1024) 2 FROM v\$logfile df JOIN v\$tablespace ts ON ts.ts# = df.ts#;</pre>				
Data_File_Num	Table Space Name	LOCATION	BYTES	MB
1	SYSTEM	C:\TEA\DISK01\SYSTEM\SYSTEM01.DBF	209715200	200
2	SYSTEM	C:\TEA\DISK04\SYSTEM\SYSTEM01.DBF	209715200	200
3	SYSAUX	C:\TEA\DISK02\SYSAUX\SYSAUX01.DBF	1153433600	1100
4	UNDOTBS	C:\TEA\DISK05\UNDOTABLE\UNDOTBS01.DBF	157286400	150
5	USERDATA	C:\TEA\DISK03\DATA\USERDATA01.DBF	52428800	50
6	SUPPLIERDATA	C:\TEA\DISK03\DATA\SUPPLIERDATA01.DBF	52428800	50
7	ADHOC	C:\TEA\DISK05\ADHOC\ADHOC01.DBF	52428800	50
8	INDX	C:\TEA\DISK04\INDX\INDEX01.DBF	52428800	50
9	READONLY	C:\TEA\DISK04\READONLY\READONLY01.DBF	52428800	50
10	COMPANYDATA	C:\TEA\DISK03\DATA\COMPANYDATA01.DBF	52428800	50
11	XDB	C:\TEA\DISK02\XDB\XDB01.DBF	734003200	700

11 rows selected.

SQL> SELECT status, name, block_size FROM v\$controlfile;		
STATUS	NAME	BLOCK_SIZE
	C:\TEA\DISK01\CONTROLFILE\CONTROLFILE01.CTL	16384
	C:\TEA\DISK02\CONTROLFILE\CONTROLFILE02.CTL	16384
	C:\TEA\DISK03\CONTROLFILE\CONTROLFILE03.CTL	16384
	C:\TEA\DISK05\CONTROLFILE\CONTROLFILE04.CTL	16384

Datafiles to Collect External to the database			
File Location	File Name	Type	
C:\app\Administrator\product\11.2.0\dbhome_1\NETWORK\ADMIN	tnsnames.ora	Network File	
	listener.ora	Network File	
	sqlnet.ora	Network File	
C:\TEA\	inittea.ora	PFILE	
C:\app\Administrator\product\11.2.0\dbhome_1\database	SPFILETEA.ORA	SPFILE	

Appendix 2 – Linux Oracle “hot backup” shell script syntax example

- Also available in the digital portfolio in .sh format

```
#!/bin/ksh
#
# Author : Kenny Vigar
# Date : April 18th 2017
#
# Purpose : Hot Backup Script
#
# Logs : /home/oracle/logs
# Scripts : /home/oracle/scripts
# Backup Location : /u04/backups/

# verify only one parameter passed to orahot.sh
if [ $# != 1 ]; then
echo " "
echo "orahot.sh needs a single parameter"
exit 1
fi
#validation of no empty parameter or more than one parameter

#do not confirm oraenv
ORAENV_ASK=NO
ORACLE_SID=$1

# $$ is process ID

TMPFILEstatus=/tmp/orahot$$_.txt
TMPFILEarchivestatus=/tmp/orahotarchive$$_.txt
TMPFILEctl=/tmp/orahotctlfile$$_.txt
TMPTrace=/u01/app/oracle/diag/rdbms/tst1/tst1/trace
TMPFILEbackup=/tmp/orahottempfilebackup$$_.txt
TEMPDBFILEGATHER=/tmp/orahotfilegather$$_.txt
TEMPARCGATHER=/home/oracle/scripts/temp/temparchgather$$_.txt

YYMMDD=`date +"%Y%m%d%H%M%S"`
YYMMDD_formatted=`date +"%Y_%m_%d %H:%M:%S"`

BU_LOCATION=/u04/backup
BU_FILES_DIR=${BU_LOCATION}/${ORACLE_SID}_${YYMMDD}

LOG_FILE=/home/oracle/log/orahot_${ORACLE_SID}_${YYMMDD}.log
#LOG_FILE=${BU_LOCATION}/${ORACLE_SID}_${YYMMDD}/${ORACLE_SID}_${YYMMDD}.log
SEQtemp1=/tmp/seqtemp1$$.txt
SEQtemp2=/tmp/seqtemp2$$.txt

RECOVERY=${BU_FILES_DIR}/recovery_${YYMMDD}.sh

#error log function -a is append
#sends log to /home/oracle/log

echo_msg()
{
    echo "orahot: $1 " | tee -a ${LOG_FILE}
```

```

}

echo_msg "-----"
echo_msg " "
echo_msg "Backup script starting"
echo_msg " "
echo_msg "Start time: ${YYMMDD}_formatted"
echo_msg " "
echo_msg "Last Modified by: Kenny Vigar"
echo_msg " "
echo_msg "This script performs a hot backup on all tablespaces in ${ORACLE_SID}"
echo_msg "and all archive log files with the sequence numbers captured during backup"
echo_msg " "
echo_msg "Backup location : ${BU_FILES_DIR}"
echo_msg "Log File : ${LOG_FILE}"
echo_msg "Script found in : /u01/home/oracle/scripts/orahot.sh"
echo_msg " "
echo_msg "-----"

#run dbhome first - check if sid is on this server.. if not - quit!
#oraenv should not prompt for home now if the home does not exist
dbhome > /dev/null 2>&1
if [ $? != 0 ]; then
echo_msg "$ORACLE_SID is not a valid instance on this server"
exit 1
fi

#send the output of oraenv to dev/null - no output
. oraenv > /dev/null 2>&1

#check return code to make sure oracle is using the proper oraenv
if [ $? = 0 ]; then
echo_msg " "
echo_msg "Environment changed to SID : ${ORACLE_SID}"
else
echo_msg "Unable to set the enviroment for SID : ${ORACLE_SID}"
exit 1
fi

#backup location
echo_msg "The backup location will be ${BU_FILES_DIR}"

#make backup location
mkdir ${BU_FILES_DIR} > /dev/null 2>&1
if [ $? != 0 ]; then
echo_msg "*** Unable to create backup location. Cancelling backup....."
exit 1
else
echo_msg "*** Backup location created. - SUCCESS"
fi

# verify database is running and redirect output to temp file
sqlplus -s /nolog <<EOF > ${TMPFILEstatus}
whenever sqlerror exit 1
connect / as sysdba

```

```

set head off
set feedback off
set trimspool off
set pagesize 0
select status as "Database Status" from v$instance;
EOF
if [ $? != 0 ]; then
echo_msg "*** Unable to verify status of database. Cancelling backup....."
exit 1
fi

#Error check on DB Status -
if [ `cat ${TMPFILEstatus}` != 'OPEN' ]; then
echo_msg "*** Database not open. Cancelling backup....."
exit 1
else
echo_msg "*** Database status is.. `cat ${TMPFILEstatus}` Success!"
fi


#verify archive log mode
sqlplus -s /nolog <<EOF > ${TMPFILEArchivestatus}
whenever sqlerror exit 1
connect / as sysdba
set head off
set feedback off
set trimspool off
set pagesize 0
select log_mode as "Log Status" from v$database;
EOF
if [ $? != 0 ]; then
echo_msg "*** Unable to verify log mode of database. Cancelling backup...."
exit 1
fi

if [ `cat ${TMPFILEArchivestatus}` != 'ARCHIVELOG' ]; then
echo_msg "*** Database not in Archive Log Mode. Cancelling backup...."
exit 1
else
echo_msg "*** Database in `cat ${TMPFILEArchivestatus}` - Continuing backup"
fi

#verify current archive sequence number

# select name, sequence# from v$archived_log where sequence# between 5 and 7 order by
first_time
# sequence check after ts is backed up

#verify archived_log reincarnation matches the v$database_incarnation

sqlplus -s /nolog <<EOF > ${SEQtemp1}
whenever sqlerror exit 1
connect / as sysdba
set head off
set feedback off
set trimspool off
set tab off

```

```

set pagesize 0
select max(sequence#) from v$log;
EOF
if [ $? != 0 ]; then
  echo_msg "*** Unable to determine current log sequence. Exiting backup...."
  exit 1
fi

currSEQ=`grep -v '^$' ${SEQtemp1} | sed 's/ //g'` 
echo_msg " "
echo_msg "*** Current sequence number ${currSEQ}"
echo_msg " "

# Begin tablespce backups
# Is anything currently in backup mode?

sqlplus -s /nolog <<EOF > ${TMPFILEbackup}
whenever sqlerror exit 1
connect / as sysdba
set head off
set feedback off
set trimspool off
set pagesize 0
select status from v$backup where status ='ACTIVE';
EOF

if [ $? != 0 ]; then
  echo_msg "*** Error checking Database files currently backup mode. Exiting backup...."
  echo_msg "*** Verify v$backup is does not have any ACTIVE backup"
exit 1
fi

if [ "`cat ${TMPFILEbackup}`" = "ACTIVE" ]; then
  echo_msg "*** Datafile already in Active Backup Mode. Exiting backup... "
  echo_msg "*** Verify v$backup is not showing any ACTIVE backup"
exit 1
else
  echo_msg "*** Verified - No tablespaces currently in backup mode... continuing backup"
fi
echo_msg " "
echo_msg "*** All checks good! - continue with backup"
echo_msg ""
echo_msg "*** Datafile gathering starting - copying to ${BU_FILES_DIR}"

#gather list of data files
sqlplus -s /nolog <<EOF > ${TEMPDBFILEGATHER}
whenever sqlerror exit 1
connect / as sysdba
set pagesize 0
set linesize 2048
set head off
set feedback off
set trimspool on
select tablespace_name ||':'||file_name||':'||${BU_FILES_DIR}///
      ||substr(file_name,instr(file_name,'/', -1,1)+1, length(file_name))///
      ||'.'||file_id

```

```

from sys.dba_data_files
order by tablespace_name;
EOF

if [ $? != 0 ]; then
echo_msg "*** Error gathering files for backup. Exiting backup....."
exit 1
fi

cat ${TEMPDBFILEGATHER} | awk -F: '{print $1 " " $2 " " $3}' | while read TBL DTF BUF
do
sqlplus -s /nolog <<EOF > /dev/null
whenever sqlerror exit 1
conn / as sysdba
alter tablespace ${TBL} begin backup;
EOF

if [ $? != 0 ]; then
echo_msg "*** Tablespace ${TBL} begin backup failed. Exiting backup....."
exit 2
fi
#cp ${DTF} ${BUF}
#touch to 'simulate' copy command that is commented out above
touch ${BUF}
#push recovery copy to recover script
echo "#cp ${BUF} ${DTF}" >> ${RECOVERY}
if [ $? != 0 ]; then
echo_msg "*** Failed to copy (touch) ${BUF}. Check permissions to destination. Exiting
backup...""
fi
sqlplus -s /nolog <<EOF > /dev/null
whenever sqlerror exit 1
conn / as sysdba
alter tablespace ${TBL} end backup;
EOF
if [ $? != 0 ]; then
echo_msg "*** ${TBL} alter tablespace end backup failed. Exiting backup...."
exit 2
else
echo_msg "*** Success backup of tablespace : ${TBL} "
fi

done

#one more archive log switch now that loop is complete

sqlplus -s /nolog <<EOF > /dev/null
whenever sqlerror exit 1
connect / as sysdba
set pagesize 0
set head off
set feedback off
set trimspool on
alter system archive log current;
EOF
if [ $? != 0 ]; then

```

```

echo_msg "*** Archive log current failed.  Exiting backup...."
exit 2
fi

#backup complete
echo_msg "*** Sucessful backup of all datafiles.  Continuing to verify archive log sequences
required for backup"
echo_msg " "
#check end sequence
#verify archived_log reincarnation matches the v$database_incarnation

sqlplus -s /nolog <<EOF > ${SEQtemp2}
whenever sqlerror exit 1
connect / as sysdba
set head off
set feedback off
set trimspool off
set tab off
set pagesize 0
select max(sequence#) from v\$archived_log;
EOF
if [ $? != 0 ]; then
echo_msg "*** Unable to determine current log sequence.. Exiting backup...."
exit 1
fi

endSEQ=`grep -v '^$' ${SEQtemp2} | sed 's/ //g'`
echo_msg " "
echo_msg "*** End sequence number ${endSEQ}"
echo_msg " "

#gather list of data files
sqlplus -s /nolog <<EOF > ${TEMPARCGATHER}
whenever sqlerror exit 1
connect / as sysdba
set pagesize 0
set linesize 2048
set head off
set feedback off
set trimspool off
set tab off

select a.name
from v\$archived_log a join v\$database_incarnation d on d.resetlogs_id = a.resetlogs_id
where a.sequence# between ${currSEQ} and ${endSEQ};
EOF
if [ $? != 0 ]; then
echo_msg "*** Unable to determine archive names. Exiting backup..."
exit 1
fi

#copy archive logs

cat ${TEMPARCGATHER} | awk -F: '{print $1}' | while read ARCLOGSOURCE
do
cp ${ARCLOGSOURCE} ${BU_FILES_DIR}
#push recovery copy to recover script

```

```

echo "#cp ${BU_FILES_DIR}/`basename ${ARCLOGSOURCE}` ${ARCLOGSOURCE}" >> ${RECOVERY}
if [ $? != 0 ]; then
  echo_msg "*** Failed to copy ${ARCLOGSOURCE} to ${BU_FILES_DIR}..... exiting.."
  exit 1
else
echo_msg " "
echo_msg "loop copy each archive log"
echo_msg "*** ${ARCLOGSOURCE} ... copied"
fi
done
echo_msg " "
echo_msg "**** Confirm files are in the backup location"
echo_msg "**** Archive Logs Copied - "
echo_msg " `ls ${BU_FILES_DIR}/*.arc`"

#backup trace/binary control files

sqlplus -s /nolog <<EOF > ${TMPFILEctl}
whenever sqlerror exit 1
connect / as sysdba
set head off
set feedback on
set trimspool off
set pagesize 0
set space 0
alter database backup controlfile to '${BU_FILES_DIR}/controlfile_${YYMMDD}.bak';
EOF

if [ $? != 0 ]; then
  echo_msg "*** Control File backup to ${BU_FILES_DIR} failed. Exiting."
  exit 1
fi

#enter copy ctl to recovery
echo "#cp ${BU_FILES_DIR}/controlfile${YYMMDD}.bak /u02/oradata/tst1/control01.ctl" >>
${RECOVERY}

#confirm control file is in the location
echo_msg " "
echo_msg "**** Binary control file successfully backed up to"
echo_msg " `ls ${BU_FILES_DIR}/con*.bak`"

sqlplus -s /nolog <<EOF > ${TMPFILEctl}
whenever sqlerror exit 1
connect / as sysdba
set head off
set feedback on
set trimspool off
set pagesize 0
set space 0
alter database backup controlfile to trace as '${BU_FILES_DIR}/controlfile_${YYMMDD}.trc';
EOF

if [ $? != 0 ]; then
  echo_msg "*** Control File backup to Trace failed. Exiting backup...."
  exit 1

```

```
fi
echo_msg " "
echo_msg "*** Trace control File successfully backed up to Trace"
echo_msg " `ls ${BU_FILES_DIR}/con*.trc`"

echo_msg " "
echo_msg "-----"
echo_msg " "
echo_msg "All datafiles successfully backed up"
YYMMDD_formatted=`date +"%Y_%m_%d %H:%M:%S"`
echo_msg "End time: ${YYMMDD_formatted}"
echo_msg " "
echo_msg "See recovery script : ${RECOVERY} "
echo_msg "See backup log      : ${LOG_FILE}"
echo_msg " "
echo_msg "Backup Complete"
echo_msg "-----"

exit
```

Travel Experts Travel Agency

Database Website Integration

February 3rd, 2017

Kenny Vigar

kenny.Vigar@edu.sait.ca

This documentation contains a proposal to develop a website front end for the Travel Experts Travel Agency. Considerations to the following will be addressed,

Website Integration with Travel Experts Travel Agency	
1	Integration Benefits
2	Existing Examples of Web Integration
3	Travel Experts Web Integration
4	Public Pages
5	Internal Pages
6	Data Validation
7	Internal User Groups and Security
8	Summary



Web Integration Benefits

Travel Experts Travel Agency (TEA) wants to guarantee solid data integrity in all of their current sales, supplier and customer records. With high expectations regarding data quality, we propose a front end graphical user interface (GUI) to support customers, agents and management.

The front end GUI will allow for data validations and a standard entry form allow agents to enter information for each client and with complete accuracy. Management can use reports, visible only to themselves with customized group assignments, to monitor their business.

Public pages will be available for customers to view new travel package specials and TEA's contact information displayed for easy communication.

The benefits of having data quality validations, such as "The customer's last name cannot be empty" will ensure records are kept accurate. The ability to enter new customer information by entry form or to check if a customer already exists will give efficient record management to agents without needing to know any 'back end' SQL code.

This front end system vastly improves upon the previous method of tracking data in Excel spreadsheets by giving a central management system to all employees. If one agent makes a change, other agents will see this change in real-time.

A comparison of competitor travel agency webpages follow, with benefits and drawbacks of each site. These comparisons have been taken into account for the design of the Travel Experts front end GUI to capitalize on the strengths and avoid weaknesses in existing competitor sites.



Welcome to Travel Experts Travel Agency. Let us book your next vacation!

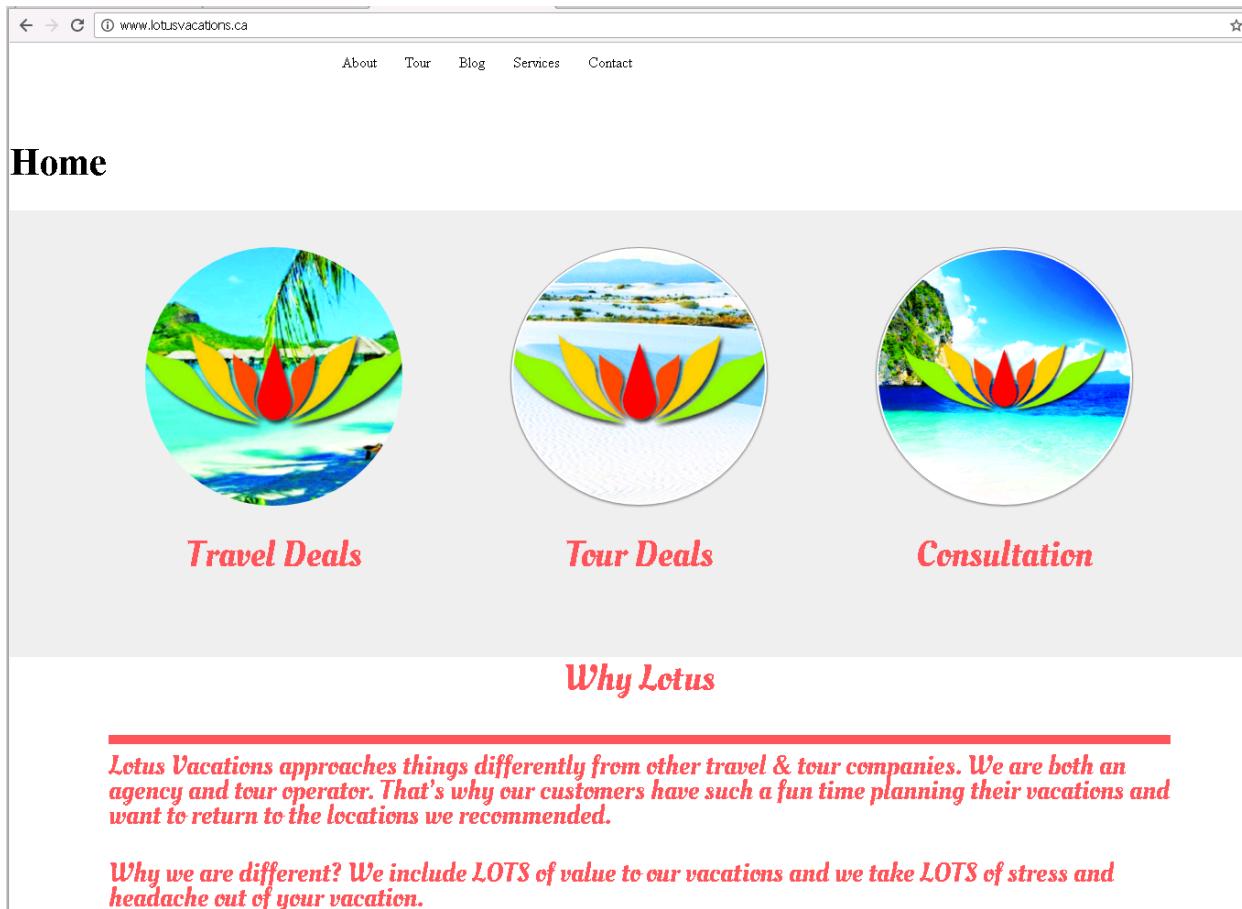


Existing Examples of Web Integration

LotusVacations.com

Lotus Vacations website has a clean background, and easy to find title links which works for a travel site like this. The contact, services, tour and about sections are top center making it easy to find what you are looking for.

Lotus Vacations would benefit by changing its font and text color. The graphics could also be a little cleaner and smaller. The Travel Experts site proposed in this document ensures a clean graphical theme and easy to read lettering.



The screenshot shows the homepage of www.lotusvacations.ca. The header includes a navigation bar with links for About, Tour, Blog, Services, and Contact. Below the header, the word "Home" is displayed in a large, bold, black font. Three circular icons are arranged horizontally, each containing a stylized lotus flower over a tropical scene. The first icon is labeled "Travel Deals" in red text below it. The second icon is labeled "Tour Deals" in red text below it. The third icon is labeled "Consultation" in red text below it. At the bottom of the page, the heading "Why Lotus" is centered above a red horizontal line. Below the line, a paragraph of text in red reads: "Lotus Vacations approaches things differently from other travel & tour companies. We are both an agency and tour operator. That's why our customers have such a fun time planning their vacations and want to return to the locations we recommended." Another red paragraph below states: "Why we are different? We include LOTS of value to our vacations and we take LOTS of stress and headache out of your vacation." A small hot air balloon icon is located in the bottom right corner of the page.



MarlinTravel.com

Marlin Travel has easy to find menus spanning the top of the webpage, similar to Lotus Vacation. The easy to find contact information is also beneficial to a client who simply wants to get a hold of an agent to book a product.

Marlin Travel's website does suffer from having far too much information on the first page. It is quite easy to 'get lost' on this page when you have package details, the contact map and promotional information grouped together.

Travel Experts Travel Agency would benefit by keeping a highlighted menu and clear information but by keeping information separate to its own relevant page. For example, contact information can be highlighted on the menu bar, but keep the map, phone numbers and address to its own entity of the website.

The screenshot shows the MarlinTravel.com homepage. At the top, there is a dark blue header with the Marlin Travel logo, a phone number (403-201-7200), an address (310 Shawville Blvd S.E., Calgary), and a 'View our Travel Professionals' button. Below the header is a light blue navigation bar with links for Vacations, Flights, Deals, Cruises, Hotels, Car Rental, Things to do, Destinations, Weddings, Book a group, AIR MILES®, and Disney. The main content area features a large banner for 'UNIWORLD BOUTIQUE RIVER CRUISE COLLECTION' with a photo of three smiling people. Below the banner, a headline reads 'Our Best Offer Ever' with a subtext 'Save up to 20% on select 2017 cruises in Europe.' A 'Best offer' button is visible. Underneath the banner is a search interface with tabs for PACKAGES, FLIGHTS, HOTELS, CAR HIRE, and CRUISES. The search fields include 'Leaving From' (Calgary - YYC), 'Going To' (Choose Your Destination), and 'Departing On' (dd/mm/yyyy). Buttons for 'More Options' and 'Start Searching' are present. At the bottom of the page is a map of the Shawnessy area in Calgary, showing streets like 162 Ave SE, 164 Ave SE, and 166 Ave SE, along with landmarks such as IHOP, Shopping Centre, and South Calgary Health Centre. A blue marker indicates the location of the travel agency's office at 310 Shawville Blvd S.E.



Flight Center's website seems more intent on getting a client into a location than providing a list of products or current specials. While this might be beneficial to capture a client in store, some clients may be turned away because of the lack of products available on the website. Using a "Public Entry" page like this runs the risk of not showing a client what you can do for them.

The screenshot shows the FlightCentre.ca homepage with a red header featuring a pilot's photo, the company name, and a phone number. Below the header is a navigation bar with links for HOME, FLIGHTS, VACATIONS, HOTELS, TOURS, CRUISES, DESTINATIONS, EXTRAS, FIND AN AGENT, DEALS, and a SEARCH bar. The main content area is titled "Find a Travel Agency Near You" and includes a search form for "Search Within 50km of calgary". To the right is a map of Calgary with 10 red pins indicating store locations, labeled 1 through 10. Each pin has a callout with the store name, address, and phone number, along with a "Store Details" button.

Store Number	Store Name	Address	Phone Number
1	Flight Centre Business Travel Bankers Hall	315 - 8 Ave S.W., Suite 156 Calgary, T2P 4K1	1 866 205 3341
2	Flight Centre The CORE	Unit 124, 751 - 3 Street S.W. Calgary, T2P 4K8	1.866.247.4117
3	Flight Centre 17th Ave	Suite 110, Mount Royal Block, 815 - 17th Avenue S.W. Calgary, T2T 0A1	1866 772 2011
4	Flight Centre Weddings and Groups at Kensington	104 - 10th Street N.W. Calgary, T2N 1V3	1 866 416 7716
5			
6			
7			
8			
9			
10			

What does work for Flight Center are the dropdown menus available. While it looks like they are not very descriptive at first glance, when you hover over them it gives you a massive amount of information regarding trips on sale, last minute vacations, vacation specials, or vendor specific vacations. The breakdown of each available vacations is nice for narrowing down what a client might want. An example of this menu follows on the next page.



Flight Center continued

The screenshot shows the Flight Centre website homepage. At the top, there's a banner with a pilot's photo and the text "Cheap flights, vacation packages and travel deals". To the right of the banner are links for "Contact Us" and "Social Ch". Below the banner is a navigation bar with tabs: HOME, FLIGHTS, VACATIONS (which is highlighted in red), HOTELS, TOURS, CRUISES, and DESTINATIONS. A dropdown menu is open under the VACATIONS tab, listing various travel options:

- Canada on Sale
- Family Vacations
- Europe on Sale
- Vacation Packages
- Last Minute Vacations
- All Inclusive Vacations
- Golf
- Ski
- Disney Parks
- Rocky Mountaineer

Below this list is a section titled "More Partners" with a "Cheap Vacation Deals" button. Underneath are more travel options:

- Toronto Vacation Deals
- Vancouver Vacation Deals
- Calgary Vacation Deals
- Ottawa Vacation Deals
- Halifax Vacation Deals
- Edmonton Vacation Deals

At the bottom left of the page, there's a "Find a Travel Agent" section with a "Search With" button and a "Flight Centre Bankers" logo. The address "315 - 8 Ave S.W, Suite 150" is also visible.

Each of the websites all have benefits which the design of Travel Experts GUI can use to stay competitive. Any of the drawbacks on these examples have been avoided in the Travel Experts design allowing you to keep an edge when it comes to clients choosing their vacation provider.

On top of capturing a client, the Travel Expert design also has an Agent login page for remote access to the database from any desk in the office to house or hotel. Agents now have the ability to work on the go!



Travel Experts Web Integration

Public Welcome Page



Starting with the Public Entry page, a prospective or returning customer can view the contact information for Travel Experts or travel packages (arrow 1 in screenshot above). The main page also shows travel agency news (arrow 3 in screenshot above) which can be updated any time Travel Experts has information to publically share.

An Agent Login is available (arrow 2 in screenshot above) to allow an agent to log in to the system for adding new customers or updating existing customer information. This component is discussed further in this document.



Public Travel Packages Page



Welcome to Travel Experts Travel Agency. Let us book your next vacation!

<input type="text"/> Q <input type="text"/> Search: Package Startdate <input type="button" value="Go"/> Actions ▾							
	Package Code	Package Name	Package Description	Package Startdate	Package Enddate	Cost	Special Cost
	1	Rocky Mountians Rail	10 day train excursion through the Alberta Rockies	06-MAY-2017	16-MAY-2017	\$2,000.00	\$1,500.00
	2	East Coast Fishing Camp	Fishing 'east coast' style. Fishing and Camping	20-MAY-2017	31-MAY-2017	\$1,500.00	\$1,199.00
	3	Egypt Pyramid	See King Tut's Tomb!! 10 days in Egypt	01-MAY-2017	10-MAY-2017	\$7,000.00	\$6,050.00
	4	Sandy beaches of Mexico	14 days in Mexico, all inclusive	01-MAY-2017	14-MAY-2017	\$2,000.00	\$1,500.00

Ability to search by start date of the trip, based off the Travel Package options

The Travel Packages menu, selected from the menu on the left of the Public Welcome Page shows travel package information pulled directly from the database. Customers can search by date, name, or sort by trip cost. This menu can be tailored to suit whichever promotions are happening at Travel Experts.

The Contact Us link on the left side of the public welcome page provides your customer with your contact information for over the phone booking or for stopping at your office to speak with an agent. The address and interactive embedded Google Map as well as your office hours are shown clearly in the example on the next page.

A full page Contact Us sample follows in the next page.



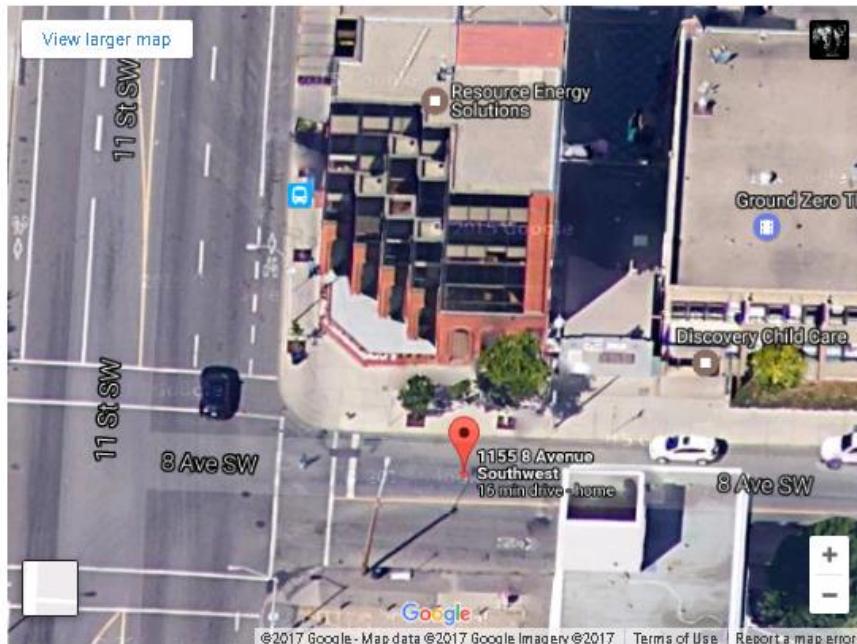
Public Contact Us Page



Welcome to Travel Experts Travel Agency. Let us book your next vacation!

We are located in downtown Calgary for your vacation booking convenience

1155 8th Ave S.W.
Calgary, AB. T2P 1N3
Ph: 403-271-9873 Fax: 403-271-9872



Please feel free to stop by any time between the business hours below

Monday - Wednesday 9AM - 5PM

Thursday 9AM - 9PM

Friday - Saturday 9AM - 5PM



Log in and Customer Records Pages



Agents and Management can login to the internal page for viewing customer records.

Log In

Username

Password

Employee logins can be controlled on a per user basis meaning only those with a secure login can access your internal system. On top of this, users can be assigned to groups to control exactly what internal documentation they would see. Management could view sales reports or employee sales figures, while Agents can enter and modify Customer information.



A new menu shows when authenticated with the front end GUI. Logout option also available.





Welcome to Travel Experts Travel Agency. Let us book your next vacation!

This page is for Employees and Agents of Travel Experts Travel Agency to manage customer details.
New customers can also be added by using the CREATE button on the bottom right.

Customer ID	First Name	Last Name	Preferred Agent
	Hiedi	Lopez	JM
	Mardig	Abdou	JD
	Ralph	Alexander	DR
	Sean	Pineda	JM
	Julita	Lippen	JD
	Pierre	Radicola	JL
	Daniel	Wicelinski	JM
	Gary	Aung	JM
	Jeff	Runyan	BD
	Omaira	Grant	DR

Create

View customer records in the database by selecting details through the pencil icon or cycle through a higher level customer report to view all records. An agent can select the blue Create button to add a record to the database.



Data Validations

Allowing employees to enter information directly enhances your productivity and ability to share knowledge around your organization. Keeping this data clean and accurate is vital to your future reporting.

1 error has occurred

- Please enter last name ([Go to error](#))

Customer Details

First Name	Kenny
Last Name *	<input type="text"/>
Preferred Agent	BD

A validation is a process which keeps a set of rules over data going into the database. In the example above the customer's last name is not entered. Before allowing this record to exist, the system is requesting the data to be as clean and accurate as it can be.

1 error has occurred

- Business Phone must be numeric. ([Row 1](#))

Customer Details

First Name	Nancy
Last Name *	Kuehn
Preferred Agent	BD

14 of 272

[Cancel](#)

CUSTOMERDETAIL Detail

	Home Phone	Business Phone	Address	City	Prov	Country	Postalcode	Email
<input type="checkbox"/>	4032693965	4032AAAAAA	44-255 9th St., SW	Calgary	AB	Canada	T1Y 6N5	

Another example of keeping data quality would be that the phone number entered must be a numeric value. If this validation does not pass, an error is raised.



Internal User Groups and Security

Custom user groups are available to you to categorize your personnel resources. Each group can have specific internal page permissions granted to them. Jr Agents may be able to view contact information for customers for marketing, while your Management Team has employee records available within their group.



Employee



Jr Agents



Management



Technical and DBA

The user authentication and security groups guarantee security in your database.

Summary and Points to Consider

The benefits of a front end GUI for Travel Experts include the following and more,

- Clean data with data validation
- Security groups
- Records management
- Public pages to share news, packages, and contact information

Your GUI's functionality is not limited by what is discussed in this proposal and additional features can be added. These may include itinerary setup, payment records, customer communications records and employee information.

It is critical to keep your data sharp for the health of the company. With minimal training to your employees, your data will be greatly enhanced.



Performance Tuning

April 10, 2017

Kenny Vigar

kvigar@gmail.com

This documentation covers all aspects of keeping a database running quickly and efficiently. A recommendation on physical (hardware) and logical (software) design and SQL code tuning to follow.

Contents	Page
Designing Performance	1
Instance Tuning	2
Memory Management	3
SQL Tuning	5
Appendix – Execution Plans	6

Designing Performance

Tuning the database is required to keep track of any high wait times to access data for either the Agents or the Apex application running on the server. Impacts of tuning can cut query time from several seconds to a fraction of a second by having reliable hardware, properly set initialization parameters, reviewing SQL query code and building performance indexes on tables.

The hardware design which the database runs on is below. The SR1100I exceeds the recommended specifications for the database.

Component	Specs	Notes
V_SR1100I Rack Server	32 GB RAM I5 E3-1220 processor - 4 threads - quadcore 4x Gigabit Ethernet LAN Ports 5x 1TB SSD HDD	- Upgraded amount of memory will keep CPU from having to keep reading from disk. - Fast quad thread, quad core processor. Fast reads when reading from disk. - Performance up!

Discussing software

Though a copy of Windows 2012R2 server is included with our hardware purchases, the database server will run Red Hat. Linux has a reputation of being very reliable. Year + server uptime and the ability to control specifically which services run on the server will keep any external interference to the database to a minimum.

Depending on the version and licensing of Oracle Software, some performance monitoring and advisor tools may not be available. This is the case with the Enterprise version of the Oracle software. In our negotiations with Oracle sales they have agreed to let us use Enterprise for the next 5 years at a significantly discounted rate then allow the option to downgrade if needed.

The following is considered regarding database design

- 3rd normal form - Data is not repeated and is not taking up additional database space because it can be referenced from its own database entity.
- Application code – Having standards for the SQL written for invoice, commission or marketing reports will be standardized to keep code in memory. This is important as any minute changes in code require the CPU to reference the disk again, not taking advantage of the code already in memory. Coding standards will also be applied to all stored procedures and functions.
- Following the best practice of running test code in a dev/test environments avoiding any rogue impact to production performance.
- Before rolling out new hardware – ‘simulate’ production work loads in test environment with Database Replay
- Indexes on primary and foreign keys. Create performance indexes on any frequently queried columns or every column in the SELECT statement (index only execution plan) if the statement is run often to avoid full table scans.

Keeping an even balance between performance and budget, without going overboard on either, was our number one priority on the plan above.

Instance Tuning

Instance Tuning refers to setting appropriate initializing parameters for the database to run. Goals are set and database parameters are configured to those specified goals. Sticking to the goal will keep you from over tuning and possibly ruining the performance gains you have already found.

Parameters which are currently set in the database are found by either running the SQL below or going through Enterprise Manager > Server > Database Configuration > Initialization Parameters.

The instance can also benefit from a Resource Plan. Consumer Groups are assigned Resource Plans to control the CPU usage, parallelism, number of active sessions and idle time. Different teams or departments within the Travel Agency can have different priorities to the server processing. To set a resource plan use

```
ALTER SYSTEM resource_manager_plan = plan_name;
```

Invalid database objects will cause code failure and should be monitored. These invalid objects will hamper database performance as well.

Monitor any corrupt database objects by querying

```
SELECT object_name, object_type  
FROM DBA_OBJECTS  
WHERE status = 'INVALID';
```

To rebuild an invalid Index use the commands below. You can also rebuild indexes in Enterprise Manager under Objects > Table. Rebuilding indexes can also be performed on valid DBA_OBJECTS to include any changed data in the table and is recommended to perform weekly (or nightly if busier) to keep table scans minimal, which again tunes the instance.

```
ALTER INDEX indexname REBUILD;
```

Database Patching

Oracle releases patches with new functionality and bug fixes regularly. CPU (critical patch updates) include security fixes and regression tests. The updates will be installed to keep the database running optimally using a tool called OPatch. Before applying a patch to the database, OPatch must be patched to the equivalent version as well.

In the event a patch is required and the database cannot be shut down an Online Patch will be done. Online patching keeps the database running, the process is quick and takes little memory.

Verify if a particular patch release can be run online with the syntax below

```
$ cd $ORACLE_HOME/OPatch  
$ OPatch query -is_online_patch patch location  
$ OPatch query patch location -all
```

Memory Management

Further instance tuning is performed by setting memory management parameters.

Automatic Memory Management (AMM) - Sets the total amount of memory allocated to the database and automatically allocates memory back and forth from the PGA and SGA. If a DBA were to execute the same tasks as the AMM they would need to set the `SGA_TARGET` (memory ceiling) and the `PGA_AGGREGATE_TARGET` (20% of the size of the database by default).

Additionally Automatic Shared Memory Management automatically manages memory allocation of all of the SGA (Shared Pool, Java Pool, Buffer Cache, Streams Pool and Large Pool) components. ASMM is automatically enabled under AMM.

Code to determine recommended SGA_TARGET

```
SELECT (SELECT SUM(value) FROM v$sga –  
       (SELECT current_size FROM v$sga_dynamic_free_memory)) "SGA Target"  
FROM dual;
```

Source: Oracle Documentation https://docs.oracle.com/cd/E18283_01/server.112/e17120/memory004.htm#1014121

ASMM and AMM also provide charted recommendations on how high to set your memory percentages within Enterprise Manager showing the ratio of available memory to performance gains. This way you will not over allocate memory – essentially wasting resources.

The recommendation to Travel Experts Travel Agency is to run AMM as opposed to manually setting the parameters above. Both can be enabled by using Enterprise Manager's Server tab > Memory Advisor.

Statistics

Cumulative statistics (wait time, time model), metrics (statistic rates) and sampled stats (stats by session, session history, stats by SQL, stats by service) can be gathered and reported on. With these reports any performance issues can be analysed and resolved. Optimizer statistics also keep track of I/O performance.

The AWR (Automatic Workload Repository) can also gather and displays memory statistics to view within Enterprise Manager. The MMOM (Manageability Monitor) background process takes a snapshot of the database stats every hour by default. Statistics can also be gathered with SQLPLUS in the `DBMS_STATS.GATHER_SYSTEM_STATS` procedure. The `SYS.AUX_STATS` table can be queried to view these.

It's worth noting that you can use statistics gathered from another database to compare against the database you are tuning. A test and production setup for example.

The Automatic Database Diagnostic Monitor (ADDM) reports top problems showing in the snapshots and reports them through Enterprise Manager.

SQL Tuning

Statistics on top running SQL or top active sessions can be analysed to determine bottlenecks in the database. SQL tuning can make the application or user queries faster and more efficient. SQL Tuning Advisor can provide the following benefits

- Identifying and tuning top SQL statements using most resources (I/O and memory) or taking the most time
- Analyzing one statement or many (tuning sets) at a time or from historical AWR information. Duplicate SQL can also be detected.
- Runs nightly against major high load SQL statements
- Use automatic tuning to view recommendations on SQL code restructure, performance index builds (access path analysis), etc.
- Validates its own statistics against itself with SQL profiling.

Access the SQL Tuning Advisor in Enterprise manager by selecting *Advisor Central* and choosing *Advisor Type*.

When a SQL Statement runs, Oracle determines the best execution plan to retrieve the data set. A cost based approach is used to build the execution plan with each part of the query composing of a ‘load’ on the plan.

Viewing this plan can be done by using the AUTOTRACE command

```
SQL> ALTER SESSION SET SQL_TRACE=true;  
SQL> set autotrace on timing on
```

Optimizing Queries

Selecting a single row by primary key or row id would bring a quicker result than a full table scan (SELECT * FROM). Join statements work more efficiently than Subquery statements and therefore have less cost in the cost based optimizer building the execution plan.

Another SQL Tuning point to note is if a Function is being used in a query, you need to build the index on the function as a “Function based Index”

```
CREATE INDEX cust_upper_idx ON customers (upper(substr(cust_first,1,1)) customer_ref);
```

Appendix – Comparison of Full Table Scan vs Index Scan - Execution Plans

Sample Execution Plan (Full Table Scan)

```
Elapsed: 00:00:00.17

Execution Plan
-----
Plan hash value: 2844954298

-----| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-----|   0 | SELECT STATEMENT   |           | 271   | 5691  |        4 (0) | 00:00:01 |
|   1 |  TABLE ACCESS FULL | CUSTOMER | 271   | 5691  |        4 (0) | 00:00:01 |

Statistics
-----
       609 recursive calls
         0 db block gets
       171 consistent gets
       19 physical reads
         0 redo size
  11125 bytes sent via SQL*Net to client
    721 bytes received via SQL*Net from client
     20 SQL*Net roundtrips to/from client
     30 sorts (memory)
       0 sorts (disk)
    271 rows processed
```

Sample Execution Plan (Index Scan from WHERE clause filter)

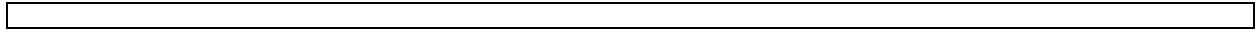
```
Elapsed: 00:00:00.06

Execution Plan
-----
Plan hash value: 2571908341

-----| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-----|   0 | SELECT STATEMENT   |           | 1     | 33   |        3 (0) | 00:00:01 |
|   1 |  TABLE ACCESS BY INDEX ROWID| SALE    | 1     | 33   |        3 (0) | 00:00:01 |
|*  2 |   INDEX RANGE SCAN    | SALE_IDX | 1     |      |        2 (0) | 00:00:01 |

Predicate Information (identified by operation id):
-----
2 - access ("PAYMENT_REF"=2339)

Statistics
-----
       0 recursive calls
       0 db block gets
       3 consistent gets
       0 physical reads
       0 redo size
  1145 bytes sent via SQL*Net to client
   523 bytes received via SQL*Net from client
     2 SQL*Net roundtrips to/from client
     0 sorts (memory)
     0 sorts (disk)
     1 rows processed
```



Invoice and Commission Scripts

April 24, 2017

Kenny Vigar

kvigar@gmail.com

A major deliverable for the Travel Experts Travel Agency Database project is to generate invoice or commission reports for agents on demand. Below is code and examples of these reports.

Contents	Page
Invoice Report Sample	1
Commission Report Sample	2
Invoice Report Script	4
Commission Report Script	7

TEA Sample Invoice Report

A sample invoice is included below. The script used to generate this is included in the appendix. This script successfully checks for all bookings under one itinerary number with PLSQL loops. Formatting has been set with RPAD functionality to ensure the spacing remains the same no matter which client information is pulled.

- Travel Experts Travel Agency - .

To: Freedom Annunziato
45 Adelphi St. #2W, NE
Calgary, AB
T2V 2K7 Bill Date: 26-JAN-14
Consultant: Jane
Customer No: 206
Passengers:
Itinerary No: 705

Prepared For:
Freedom Annunziato

Trip Details:
01 March 2014
04 March 2014
Asia

Supplier	Description	Booking No.	Start Date	End Date	Price
UNITED AIRLINES ATLANTIC BLUE CROSS	Airlines Travel Insurance	DFWSE2 KKJH243	10-JAN-14 01-MAR-14	15-JAN-14 04-MAR-14	\$900.00 \$250.00

For your convenience the following rewards were applied

Freedom Annunziato
Aero Plan 123546889

Trip Total
Sub Total: \$1,150.00
Booking Fee: \$25.00
Total: \$1,175.00

Credit Card used for Purchase: 87590432908754

TEA Sample Commission Reports

A sample commission report is included below. The script used to generate this is included in the appendix. This script successfully checks for the CANCELLED, PAID, PENDING and OVERDUE commission statuses. Formatting has been set with RPAD functionality to ensure the spacing remains the same no matter which client information is pulled

Travel Experts Travel Agency Commission Status Report

Commission Report - CANCELLED Commissions

Company Name	Commission Due Date	Original Prod End Date	Commission Amount	
Air Canada	May 15 2015	Jul 14 2015	\$60.00	CANCELLED
Celebrity Cruises	Sep 20 2014	Nov 19 2014	\$71.00	CANCELLED
United Airlines	Feb 14 2014	Apr 15 2014	\$33.00	CANCELLED
United Airlines	Dec 03 2014	Feb 01 2015	\$41.00	CANCELLED
United Airlines	Aug 29 2014	Oct 28 2014	\$38.00	CANCELLED

4.3 Cancelled Commission Report 1

Travel Experts Travel Agency Commission Status Report

Commission Report - Paid Commissions

Company Name	Commission Due Date	Original Prod End Date	Commission Amount	
Air Canada	May 13 2015	Jul 12 2015	\$35.00	PAID
Celebrity Cruises	May 31 2014	Jul 30 2014	\$76.00	PAID
National Car Rental	Apr 16 2014	Jun 15 2014	\$28.00	PAID
Royal Caribbean International	Jun 14 2014	Aug 13 2014	\$257.00	PAID
Saskatchewan Blue Cross	Apr 17 2014	Jun 16 2014	\$6.00	PAID
United Airlines	Sep 27 2014	Nov 26 2014	\$48.00	PAID
United Airlines	Jan 09 2015	Mar 10 2015	\$135.00	PAID
Western Vacations	Apr 15 2014	Jun 14 2014	\$179.00	PAID

4.4 Paid Commission Report

Travel Experts Travel Agency Commission Status Report

Commission Report - PENDING Commissions

Company Name	Commission Due Date	Original Prod End Date	Commission Amount	
Aces Aerolineas Centrales De Colombia	Jun 21 2014	Aug 20 2014	\$70.00	OVERDUE
Air Canada	Jul 29 2016	Sep 27 2016	\$25.00	OVERDUE
Air Canada	Jul 28 2013	Sep 26 2013	\$32.00	OVERDUE
Alamo Rent A Car	Oct 08 2013	Dec 07 2013	\$6.00	OVERDUE
Celebrity Cruises	Jun 13 2014	Aug 12 2014	\$251.00	OVERDUE
Compagnia Italiana Turismo Inc	Jun 24 2014	Aug 23 2014	\$169.00	OVERDUE
Eldertreks	Jun 22 2014	Aug 21 2014	\$210.00	OVERDUE

4.4 Paid Commission Report

Travel Experts Travel Agency Commission Status Report

Commission Report - PENDING Commissions

Company Name	Commission Due Date	Original Prod End Date	Commission Amount	
Air Bc	Jun 09 2016	Aug 08 2016	\$10.00	PENDING
Air Canada	Feb 28 2013	Apr 29 2013	\$60.00	PENDING
Air Canada	Feb 22 2013	Apr 23 2013	\$60.00	PENDING
Alitours Car Rental By Hertz	Oct 08 2013	Dec 07 2013	\$5.00	PENDING
Chateaux & Hotels De France	Sep 27 2014	Nov 26 2014	\$62.00	PENDING
Embassy Suites - Toronto/Markham	Oct 08 2013	Dec 07 2013	\$8.00	PENDING
Outrigger Hotels & Resorts	Apr 16 2014	Jun 15 2014	\$10.00	PENDING
United Airlines	Jan 26 2014	Mar 27 2014	\$38.00	PENDING
United Airlines	Sep 01 2014	Oct 31 2014	\$38.00	PENDING
United Airlines	Sep 29 2014	Nov 28 2014	\$75.00	PENDING
Universe Assistance	Apr 02 2014	Jun 01 2014	\$4.00	PENDING

```

-- Invoice Creation Script
-- Version 3
-- Kenny Vigar
-- Used for generating invoice reports based on Itinerary Number

SET SERVEROUTPUT ON
DECLARE
  v_itinerary import number(5):= &itinerary;
  v_cust_f_name customer.first_name%type;
  v_cust_l_name customer.last_name%type;
  v_cust_address customerdetail.address%type;
  v_cust_city customerdetail.city%type;
  v_cust_prov customerdetail.prov%type;
  v_cust_postalcode customerdetail.postalcode%type;
  v_bill_date payment.bill_date%type;
  v_bill_des payment.bill_des%type;
  v_agent_name employee.first_name%type;
  v_cust_ref customer.cust_ref%type;
  v_num_pass itinerary.num_travellers%type;
  v_itinerary_ref itinerary.itinerary_ref%type;
  v_itin_start itinerary.prod_start%type;
  v_itin_end itinerary.prod_end%type;
  v_destination destination.dest_des%type;
  v_prod_des itinerary.product%type;
  v_supplier_name supplier.company_name%type;
  v_base_price itinerary.base_price%type;
  v_booking_ref itinerary.booking_ref%type;
  v_rewards customerdetail.rewards_number%type;
  v_rewards_type customerdetail.rewards_type%type;
  v_creditcard_usetopay payment.creditcard_usetopay%type;
  v_subtotal number(5);
  v_fees number(5);
  v_total number(5);
  -- cursors

CURSOR c_customer_info IS
  SELECT c.first_name, c.last_name, cd.address, cd.city, cd.prov, cd.postalcode,
         pay.bill_date, pay.bill_des, e.first_name, c.cust_ref, i.num_travellers, i.itinerary_ref, cd.rewards_number, cd.rewards_type,
         pay.creditcard_usetopay
    FROM customer c join customerdetail cd on c.cust_ref=cd.cust_ref
      join sale sa on c.cust_ref = sa.cust_ref
        join payment pay on sa.payment_ref = pay.payment_ref
        join employee e on sa.agent_ref = e.agent_id
        join itinerary i on sa.booking_ref = i.booking_ref and i.itinerary_ref = sa.itinerary_ref
   WHERE i.itinerary_ref = v_itinerary_import;

CURSOR c_trip_details IS
  SELECT i.prod_start, i.prod_end, d.dest_des
    FROM itinerary i join destination d on i.destination_id = d.dest_id
   WHERE i.itinerary_ref = v_itinerary_import;

```

```

CURSOR c_booking_info IS
  SELECT s.company_name, i.product, i.booking_ref, i.prod_start, i.prod_end, base_price
    FROM supplier s join product p on s.supplier_ref = p.supplier_ref and s.region_ref = p.region_ref
                      join itinerary i on i.product_cat = p.product_cat and i.region_ref = p.region_ref and p.supplier_ref =
i.supplier_ref
   WHERE i.itinerary_ref = v_itinerary_import ;

BEGIN
  OPEN c_customer_info;
  DBMS_OUTPUT.PUT_LINE(chr(10));
  DBMS_OUTPUT.PUT_LINE(chr(10));
  DBMS_OUTPUT.PUT_LINE(chr(10));
  DBMS_OUTPUT.PUT_LINE('*****');
  DBMS_OUTPUT.PUT_LINE('*****');
  DBMS_OUTPUT.PUT_LINE(chr(10)||RPAD(' ',25)||'. - Travel Experts Travel Agency - .');

LOOP
  FETCH c_customer_info INTO
    v_cust_f_name, v_cust_l_name, v_cust_address, v_cust_city, v_cust_prov, v_cust_postalcode, v_bill_date, v_bill_des,
v_agent_name,
    v_cust_ref, v_num_pass, v_itinerary_ref, v_rewards, v_rewards_type, v_creditcard_usetopay;
  EXIT WHEN c_customer_info%NOTFOUND;
END LOOP;
  close c_customer_info;

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('To: ',48)||' '||chr(9)||'Bill Date: '||v_bill_date);
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_f_name||' '||v_cust_l_name,48)||'Consultant: '||v_agent_name);
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_address,48)||'Customer No: '||v_cust_ref);
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_city||', '||v_cust_prov, 48)||'Passengers: '||v_num_pass);
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_postalcode, 48)||'Itinerary No: '||v_itinerary_ref);
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('Prepared For: '); -- need a new cursor for prepared for - with loop if using traveller table
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_cust_f_name||' '||v_cust_l_name, 48));
DBMS_OUTPUT.PUT_LINE(chr(10));

  OPEN c_trip_details;
LOOP
  FETCH c_trip_details INTO
    v_itin_start, v_itin_end, v_destination;

EXIT WHEN c_trip_details%NOTFOUND;
END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('Trip Details: ');
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(to_char(v_itin_start, 'DD Month YYYY'), 48));
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(to_char(v_itin_end, 'DD Month YYYY'), 48));
DBMS_OUTPUT.PUT_LINE(chr(9)||RPAD(v_destination, 48));

```

```

DBMS_OUTPUT.PUT_LINE(chr(10));

    CLOSE c_trip_details;

OPEN c_booking_info;

DBMS_OUTPUT.PUT_LINE(RPAD('Supplier', 20) || RPAD('Description',20) || RPAD('Booking No.',20) || RPAD('Start Date',20) || RPAD('End Date',20) || RPAD('Price',20));
DBMS_OUTPUT.PUT_LINE(chr(10));
LOOP
    FETCH c_booking_info INTO
        v_supplier_name, v_prod_des, v_booking_ref, v_itin_start, v_itin_end, v_base_price;

    EXIT WHEN c_booking_info%NOTFOUND;

DBMS_OUTPUT.PUT_LINE(RPAD(v_supplier_name ,20) || RPAD(v_prod_des,20) || RPAD(v_booking_ref, 20) || RPAD(v_itin_start,
20) || RPAD(v_itin_end, 20) || RPAD((to_char(v_base_price, '$99,999.99')),20));

END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

DBMS_OUTPUT.PUT_LINE(chr(10)||'For your convenience the following rewards were applied');
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10)||chr(9)||v cust f name||' '||v cust l name);
DBMS_OUTPUT.PUT_LINE(chr(9)||chr(9)||chr(9)||RPAD(v_rewards_type, 10)||v_rewards);

CLOSE c_booking_info;

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('Trip Total');

SELECT SUM(i.base_price) subtotal, SUM(p.fee_amt) fees, SUM(i.base_price) + SUM(p.fee_amt) total into v_subtotal, v_fees, v_total
FROM itinerary i JOIN sale s ON i.booking_ref = s.booking_ref and i.itinerary_ref = s.itinerary_ref
JOIN payment p ON s.payment_ref = p.payment_ref
WHERE i.itinerary_ref = v_itinerary_import;

DBMS_OUTPUT.PUT_LINE(chr(9)||'Sub Total: ' ||LPAD((to_char(v_subtotal, '$99,999.99')),20));
DBMS_OUTPUT.PUT_LINE(chr(9)||'Booking Fee: ' ||LPAD((to_char(v_fees, '$99,999.99')),18));
DBMS_OUTPUT.PUT_LINE(chr(9)||'Total: '||LPAD((to_char(v_total, '$99,999.99')),24));

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(9)||'Credit Card used for Purchase: ' || v_creditcard_usetopay);

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('*****');
END;
/

```

```

-- Commission Reporting
-- Version 1

-- Kenny Vigar
-- Used for generating Commission Reports

col company_name format a30
col product_des format a30

SET SERVEROUTPUT ON

-- Declare variables and cursors and exception

DECLARE
  v_commission_status varchar2(10) := UPPER(TO_CHAR('&commission_status'));
  v_company_name supplier.company_name%type;
  v_supplier_ref supplier.supplier_ref%type;
  v_region_ref supplier.region_ref%type;
  v_product_des product.product_des%type;
  v_itinerary_ref itinerary.itinerary_ref%type;
  v_booking_ref itinerary.booking_ref%type;
  v_jtin_end itinerary.prod_end%type;
  v_comm_amt sale.commission_amt%type;
  v_comm_status sale.commission_status%type;
  v_comm_due_date date;

err_comm_status EXCEPTION;

CURSOR c_paid_commission IS
SELECT s.company_name, s.supplier_ref, s.region_ref, p.product_des,
       i.itinerary_ref, i.booking_ref, i.prod_end, i.prod_end+60, sa.commission_amt, sa.commission_status
  FROM supplier s JOIN product p ON s.supplier_ref = p.supplier_ref AND s.region_ref = p.region_ref
    JOIN itinerary i ON p.supplier_ref = i.supplier_ref AND p.region_ref = i.region_ref AND i.product_cat = p.product_cat
    JOIN sale sa ON i.booking_ref = sa.booking_ref AND i.itinerary_ref = sa.itinerary_ref
 WHERE sa.commission_status = 'PAID';

CURSOR c_pending_commission IS

```

```

SELECT s.company_name, s.supplier_ref, s.region_ref, p.product_des,
       i.itinerary_ref, i.booking_ref, i.prod_end, i.prod_end+60, sa.commission_amt, sa.commission_status
  FROM supplier s JOIN product p ON s.supplier_ref = p.supplier_ref AND s.region_ref = p.region_ref
    JOIN itinerary i ON p.supplier_ref = i.supplier_ref AND p.region_ref = i.region_ref AND i.product_cat = p.product_cat
    JOIN sale sa ON i.booking_ref = sa.booking_ref AND i.itinerary_ref = sa.itinerary_ref
 WHERE sa.commission_status = 'PENDING';

CURSOR c_cancelled_commission IS
SELECT s.company_name, s.supplier_ref, s.region_ref, p.product_des,
       i.itinerary_ref, i.booking_ref, i.prod_end, i.prod_end+60, sa.commission_amt, sa.commission_status
  FROM supplier s JOIN product p ON s.supplier_ref = p.supplier_ref AND s.region_ref = p.region_ref
    JOIN itinerary i ON p.supplier_ref = i.supplier_ref AND p.region_ref = i.region_ref AND i.product_cat = p.product_cat
    JOIN sale sa ON i.booking_ref = sa.booking_ref AND i.itinerary_ref = sa.itinerary_ref
 WHERE sa.commission_status = 'CANCELLED';

CURSOR c_overdue_commission IS
SELECT s.company_name, s.supplier_ref, s.region_ref, p.product_des,
       i.itinerary_ref, i.booking_ref, i.prod_end, i.prod_end+60, sa.commission_amt, sa.commission_status
  FROM supplier s JOIN product p ON s.supplier_ref = p.supplier_ref AND s.region_ref = p.region_ref
    JOIN itinerary i ON p.supplier_ref = i.supplier_ref AND p.region_ref = i.region_ref AND i.product_cat = p.product_cat
    JOIN sale sa ON i.booking_ref = sa.booking_ref AND i.itinerary_ref = sa.itinerary_ref
 WHERE sa.commission_status = 'OVERDUE';

-- begin
-- paid

BEGIN

DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE('Travel Experts Travel Agency Commission Status Report');
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

IF v_commission_status = TO_CHAR('PAID') THEN

    DBMS_OUTPUT.PUT_LINE(' Commission Report - Paid Commissions');

END IF;

```

```

        DBMS_OUTPUT.PUT_LINE('-----');
        DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('Company Name', 30) || RPAD('Commission Due Date',25) || RPAD('Original Prod End Date',25) || RPAD('Commission
Amount',40));
OPEN c_paid_commission;
LOOP
    FETCH c_paid_commission INTO v_company_name, v_supplier_ref, v_region_ref, v_product_des, v_itinerary_ref,
    v_booking_ref, v_itin_end, v_comm_due_date, v_comm_amt, v_comm_status ;
    EXIT WHEN c_paid_commission%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE(rpad(initcap(v_company_name), 35) || rpad(to_char(v_itin_end, 'Mon DD YYYY'),
25) || rpad(to_char(v_comm_due_date, 'Mon DD YYYY'),20)
        || RPAD((to_char(v_comm_amt, '$99,999.99')),24) || v_comm_status);
END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

CLOSE c_paid_commission;

ELSIF v_commission_status = TO_CHAR('CANCELLED') THEN
    DBMS_OUTPUT.PUT_LINE(' Commission Report - CANCELLED Commissions');
    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('Company Name', 30) || RPAD('Commission Due Date',25) || RPAD('Original Prod End Date',25) || RPAD('Commission
Amount',40));
OPEN c_cancelled_commission;
LOOP
    FETCH c_cancelled_commission INTO v_company_name, v_supplier_ref, v_region_ref, v_product_des, v_itinerary_ref,
    v_booking_ref, v_itin_end, v_comm_due_date, v_comm_amt, v_comm_status ;
    EXIT WHEN c_cancelled_commission%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE(rpad(initcap(v_company_name), 35) || rpad(to_char(v_itin_end, 'Mon DD YYYY'),
25) || rpad(to_char(v_comm_due_date, 'Mon DD YYYY'),20)
        || RPAD((to_char(v_comm_amt, '$99,999.99')),24) || v_comm_status);
END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

```

```

CLOSE c_cancelled_commission;

ELSIF v_commission_status = TO_CHAR('PENDING') THEN
    DBMS_OUTPUT.PUT_LINE(' Commission Report - PENDING Commissions');
    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('Company Name', 30) || RPAD('Commission Due Date',25) || RPAD('Original Prod End Date',25) || RPAD('Commission Amount',40));
OPEN c_pending_commission;
LOOP
    FETCH c_pending_commission INTO v_company_name, v_supplier_ref, v_region_ref, v_product_des, v_itinerary_ref,
    v_booking_ref, v_itin_end, v_comm_due_date, v_comm_amt, v_comm_status ;
    EXIT WHEN c_pending_commission%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE(rpad(initcap(v_company_name), 35) || rpad(to_char(v_itin_end, 'Mon DD YYYY'),
25) || rpad(to_char(v_comm_due_date, 'Mon DD YYYY'),20)
        || RPAD((to_char(v_comm_amt, '$99,999.99')),24) || v_comm_status);
END LOOP;
DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(chr(10));

CLOSE c_pending_commission;

ELSIF v_commission_status = TO_CHAR('OVERDUE') THEN
    DBMS_OUTPUT.PUT_LINE(' Commission Report - PENDING Commissions');
    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE(chr(10));
DBMS_OUTPUT.PUT_LINE(RPAD('Company Name', 30) || RPAD('Commission Due Date',25) || RPAD('Original Prod End Date',25) || RPAD('Commission Amount',40));
OPEN c_overdue_commission;
LOOP
    FETCH c_overdue_commission INTO v_company_name, v_supplier_ref, v_region_ref, v_product_des, v_itinerary_ref,
    v_booking_ref, v_itin_end, v_comm_due_date, v_comm_amt, v_comm_status ;
    EXIT WHEN c_overdue_commission%NOTFOUND;

```

```
DBMS_OUTPUT.PUT_LINE(rpad(initcap(v_company_name), 35)||rpad(to_char(v_itin_end, 'Mon DD YYYY'),  
25)||rpad(to_char(v_comm_due_date, 'Mon DD YYYY'),20)  
||RPAD((to_char(v_comm_amt, '$99,999.99')),24)||v_comm_status);  
END LOOP;  
DBMS_OUTPUT.PUT_LINE(chr(10));  
DBMS_OUTPUT.PUT_LINE(chr(10));  
  
CLOSE c_overdue_commission;  
  
END IF;  
  
EXCEPTION  
WHEN err_comm_status THEN  
DBMS_OUTPUT.PUT_LINE('** Error ** Not a Valid Commission Status');  
  
END;  
  
/
```



Elcaro Development

Travel Experts Travel Agency

Project Team:	Jay Babbar	– Lead Database Designer
	Kenny Vigar	– Lead Data Specialist
	Shameek Sharma	– Lead Data Modeler
	Andy Gu	– Security and Permissions Expert

A combined total of 60 years experience!

PROJECT OUTLINE

- Requirement Gathering
- Database Creation
- Table Creation and Data Loading
- User Management
- Where we have Been!
- Where we will Go!



REQUIREMENT GATHERING AND DATA MODELING

Oct 2016

CREATING THE DATABASE

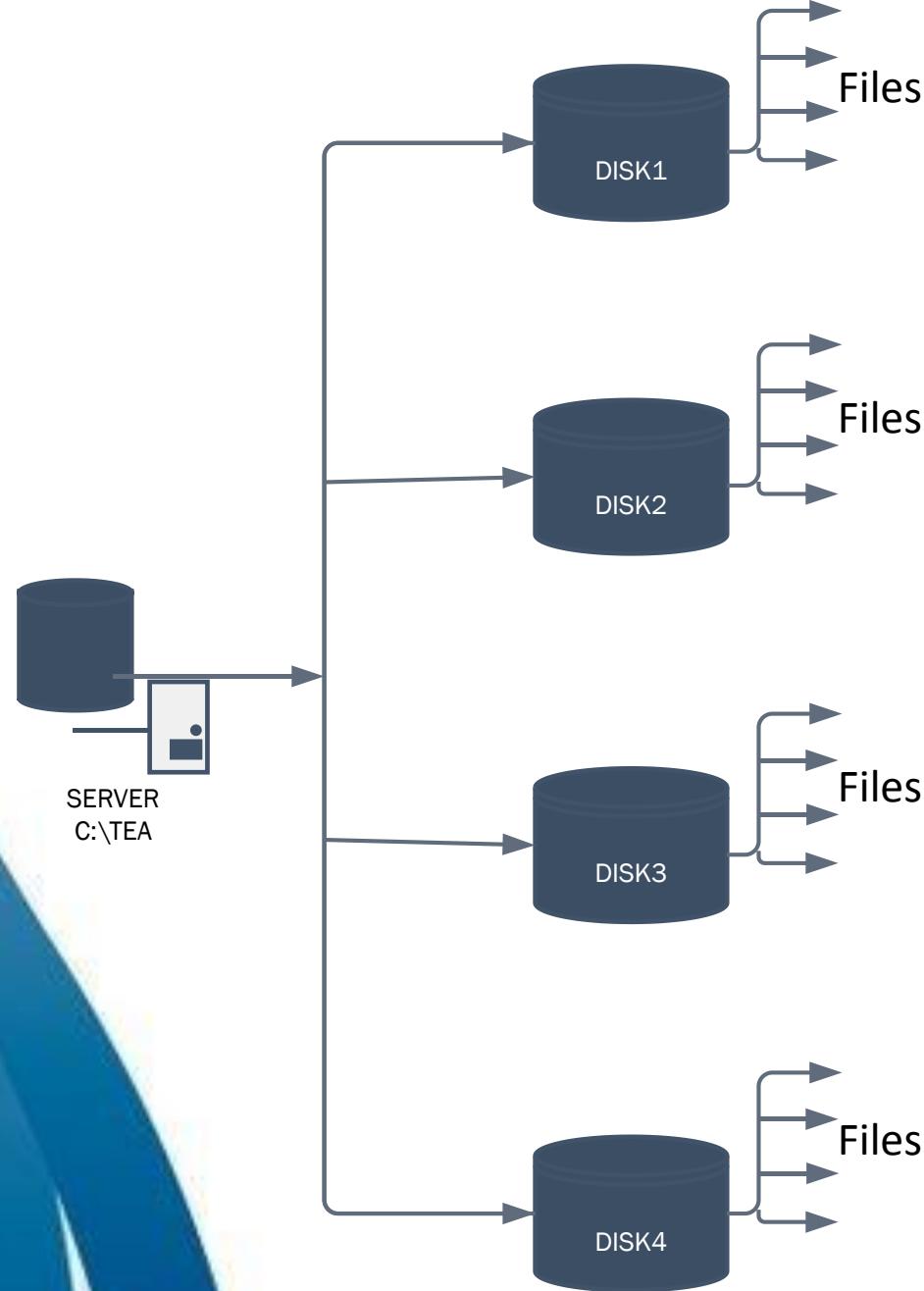
Nov 2016

Creating The Database

- Finalize the Parameters
 - Create tablespaces, control files etc
- Create the database
 - Run scripts

Where we have been!

Your Database Structure

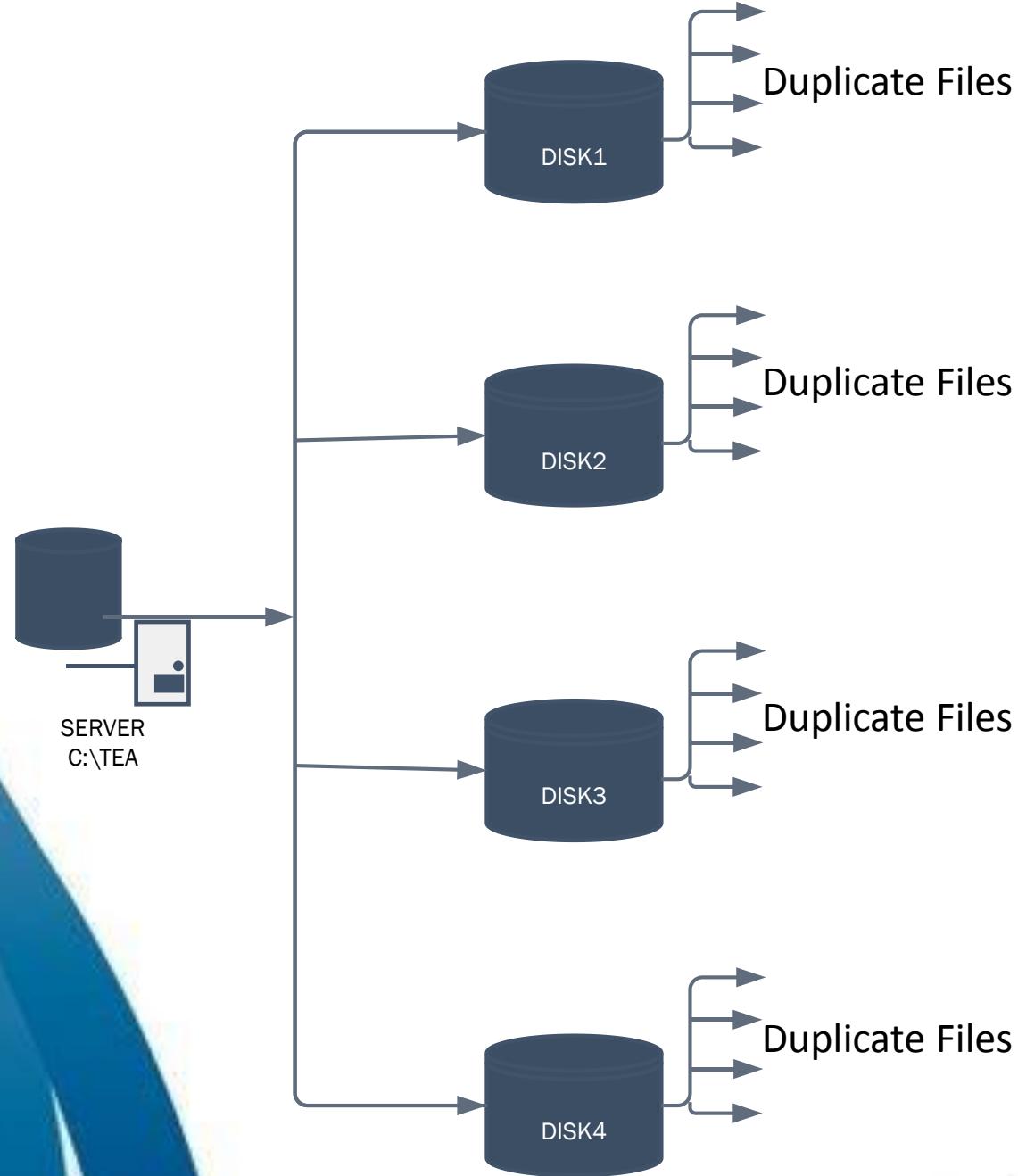


Physical Database Structure

- Database Server
- Four Disks
- All files will spread into four disks
- Design Incorporates recovery operations

Where we have been!

Your Database Safety



Data Protection and Availability

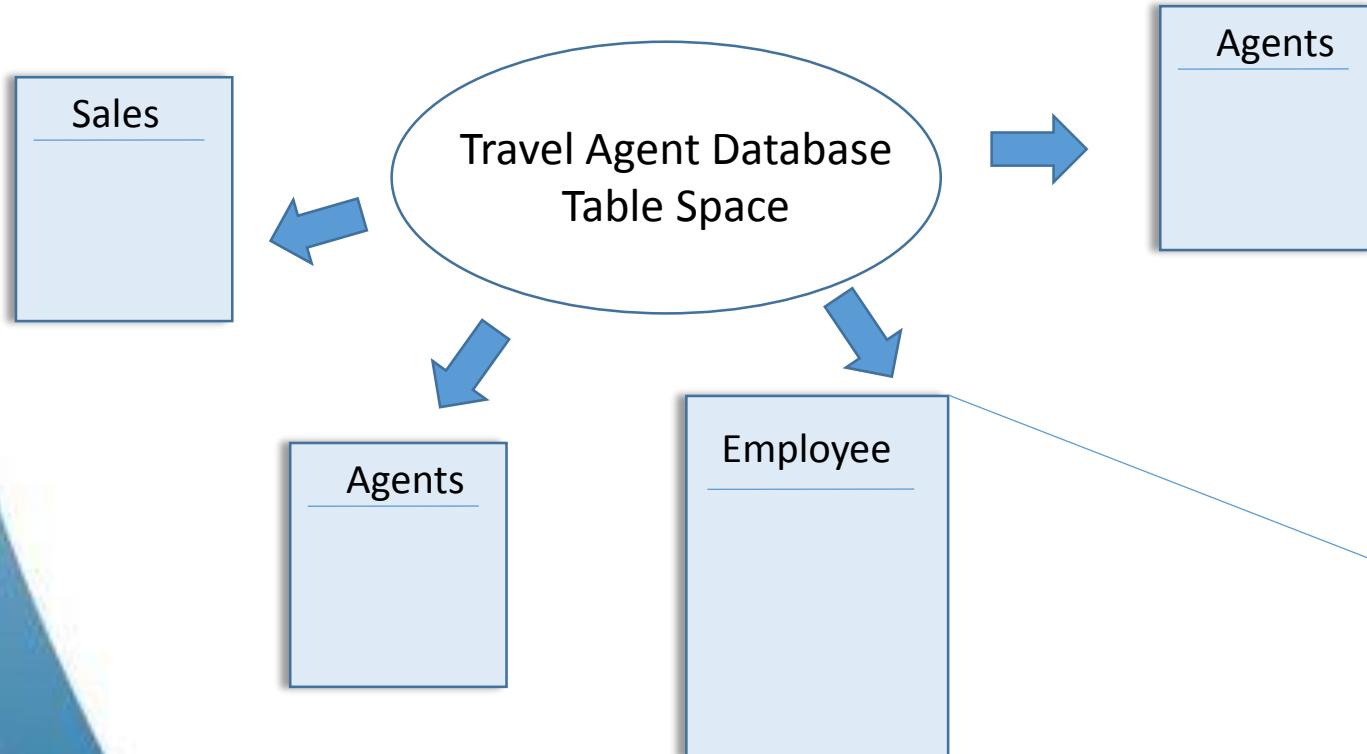
- Multiplexed files in all four disks
- Data Protection
- High Availability



CREATING TABLES AND LOADING DATA

Nov 2016

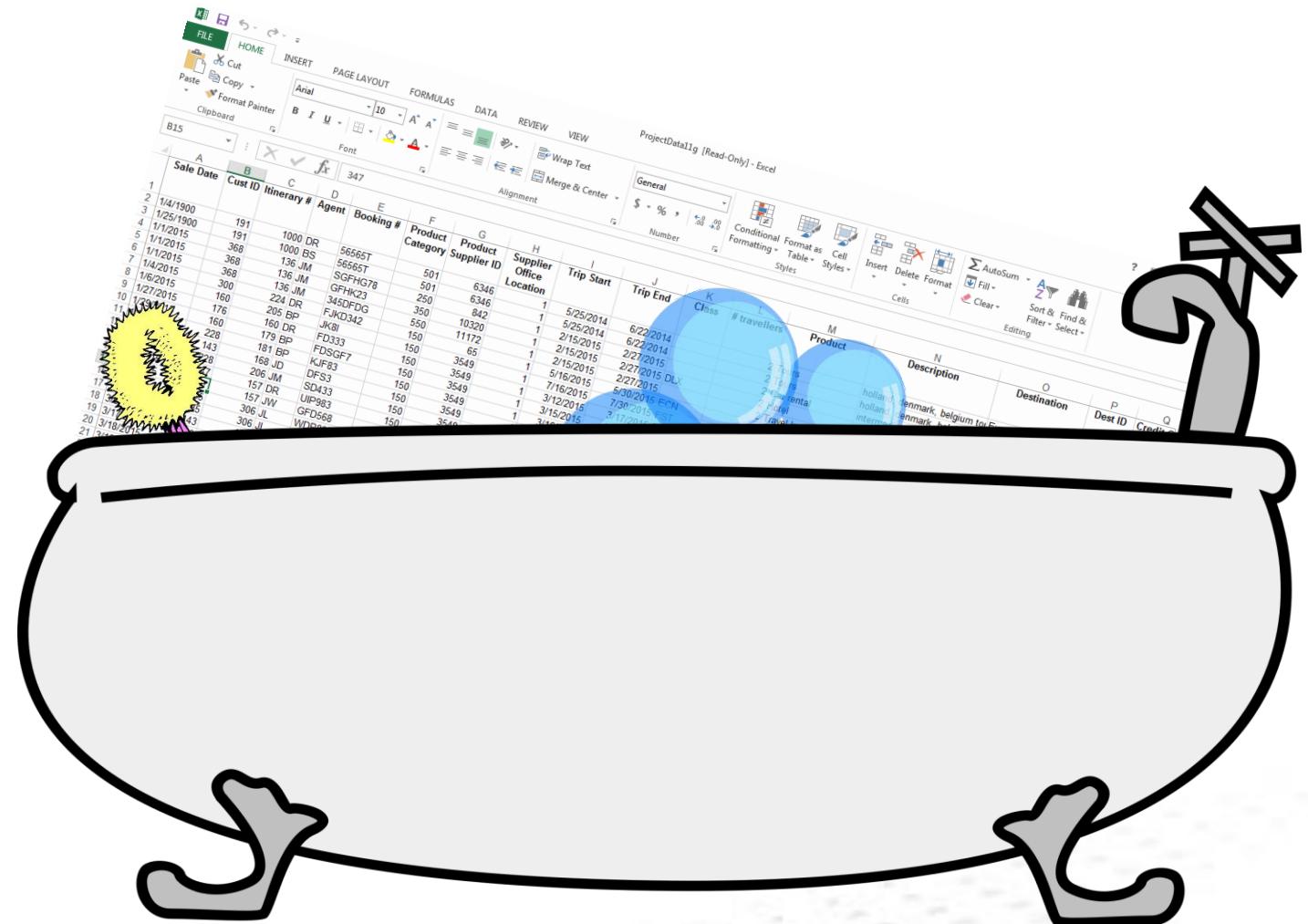
Where we have been!
Prep the Tables



```
-- Employee
CREATE TABLE employee|
  (employee_id number(3) CONSTRAINT emp_pk PRIMARY KEY deferrable initially deferred
  USING INDEX (
    CREATE INDEX emp_idx ON Employee(employee_id) PCTFREE 10 INITTRANS 2
    STORAGE (INITIAL 100k NEXT 50k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 5 )
    TABLESPACE INDX),
  first_name varchar2(20),
  last_name varchar2(20) CONSTRAINT lastname_nn NOT NULL)
  TABLESPACE userdata
  PCTFREE 20
  PCTUSED 60
  INITTRANS 2
  STORAGE (INITIAL 100k NEXT 100k MINEXTENTS 1 MAXEXTENTS 40 PCTINCREASE 10);
  COMMENT ON TABLE teaadmin.employee IS 'TEA Employees PCTFree 20, InitTrans 2, PCTUsed 60, Initial 100k, next 100k, min/max 1/40';
```

Where we have been!
Data Cleanup

Recognized the need for clean data!



Where we have been! Your Database

ProjectData11g (Read-Only) - Excel

Sale Date	Cust ID	Itinerary #	Agent	Booking #	Product Category	Product Supplier ID	Supplier Location	Trip Start	Trip End	Class	# travellers	Product	Description	Destination	Dest ID	Credit C
1 10/1/1990	191	3000 DR	6466ST	601	EMI	1	5/25/2014	6/22/2014	2 Train	2	Holland, Denmark, Belgium to Europe	EU	AMEA			
1 10/2/1990	191	3000 DS	6566ST	601	EMI	1	5/25/2014	6/22/2014	2 Train	2	Holland, Denmark, Belgium to Europe	EU	AMEA			
1 1/25/1990	368	136 JM	SFGHG78	250	B42	1	2/15/2015	2/27/2015	2 Car rental	intermediate car	Paris, France	EU	VISA			
8 1/1/2015	368	136 JM	SPHNG78	350	1020	1	2/15/2015	2/27/2015	2 Hotel	2	Paris, France	EU	VISA			
8 1/1/2015	368	136 JM	SPHNG78	350	1172	1	2/15/2015	2/27/2015	2 Hotel	2	Paris, France	EU	VISA			
7 1/4/2015	368	224 DR	FJKD342	150	65	1	5/16/2015	5/30/2015	2 Air	Calgary/Cape Town/Calgary, Cape Town, South Africa	Africa	MEAST				
10 1/1/2015	191	3000 DR	6466ST	601	EMI	1	5/25/2014	6/22/2014	2 Train	2	Holland, Denmark, Belgium to Europe	EU	AMEA			
9 1/27/2015	171	160 DR	FD313	150	3549	1	3/12/2015	3/17/2015	1 Air	Calgary/Vancouver/Calgary, Cairo, Egypt	Africa	MEAST	Diners			
10 1/29/2015	160	179 BP	FGS6FT	150	3549	1	3/15/2015	3/20/2015	1 Air	Calgary/Toronto/Calgary, Cairo, Egypt	Africa	MEAST	Diners			
11 1/29/2015	229	181 DR	FGS6FT	150	3549	1	3/15/2015	3/20/2015	1 Air	Calgary/Toronto/Calgary, Cairo, Egypt	Africa	MEAST	Diners			
12 1/31/2015	143	168 JD	DP33	150	3549	1	3/17/2015	3/22/2015	1 Air	Calgary/Vancouver/Calgary, Vancouver	NA	Diners				
13 2/6/2015	228	206 JM	SD433	150	3549	1	7/17/2015	7/31/2015	1 Air	Calgary/Carto/Calgary, Cairo, Egypt	Africa	MEAST	Diners			
13 2/6/2015	228	207 JM	SD433	150	3549	1	7/17/2015	7/31/2015	1 Air	Calgary/Carto/Calgary, Cairo, Egypt	Africa	MEAST	Diners			
13 [20/26/2015]	167	JW	GP0568	352	9766	1	4/12/2015	4/24/2015	2 Hotel	2 Hotel	Miami, Florida	NA	VISA			
18 3/6/2015	150	306 JL	WD9898	500	9396	1	5/9/2015	6/3/2015	2 Air	all inclusive European tour	London, England	EU	VISA			
17 3/6/2015	135	300 DR	WD9898	500	9396	1	5/9/2015	6/3/2015	2 Air	all inclusive European tour	London, England	EU	VISA			
18 3/6/2015	142	178 BP	FE33	150	3549	1	7/18/2015	8/1/2015	1 Air	Calgary/Calcutta/Calgary, Calcutta, India	Asia	MEAST	Diners			
19 3/9/2015	169	180 DR	FP31	150	3549	1	5/2/2015	5/27/2015	1 Air	Calgary/Vancouver/Calgary, Paris, France	EU	MEAST	Diners			
20 3/18/2015	167	309 JV	8393899	350	3773	1	5/2/2015	5/27/2015	2 Hotel	European train pass	Paris, France	EU	MEAST	Diners		
21 3/18/2015	260	317 JC	KJF9899	455	828	1	5/1/2015	5/26/2015	1 Air	Calgary/Vancouver/Calgary, Paris, France	EU	MEAST	Diners			
22 3/19/2015	331	170 DR	FP31	150	3549	1	5/2/2015	5/27/2015	1 Air	Calgary/Vancouver/Calgary, Paris, France	EU	MEAST	Diners			
23 3/19/2015	364	414 BD	FJKD343	150	3549	1	5/1/2015	5/11/2015	2 Air	Calgary/Cape Town/Calgary, Kathmandu, Nepal	ASIA	MEAST	Diners			
24 3/20/2015	202	215 DR	UP986	150	3549	1	5/3/2015	5/22/2015	1 Air	Calgary/Kathmandu/Calgary, Kathmandu, Nepal	ASIA	MEAST	Diners			
25 3/20/2015	331	216 DR	UP982	150	3549	1	5/4/2015	5/23/2015	2 Air	Calgary/Cape Town/Calgary, Kathmandu, Nepal	ASIA	MEAST	Diners			
26 3/20/2015	331	166 DR	UP982	150	3549	1	5/4/2015	5/29/2015	2 Air	Calgary/Cape Town/Calgary, Kathmandu, Nepal	ASIA	MEAST	Diners			
27 3/20/2015	364	963	FJKD343	150	3549	1	5/1/2015	5/11/2015	2 Air	Calgary/Cape Town/Calgary, Kathmandu, Nepal	ASIA	MEAST	Diners			
27 3/20/2015	364	963	FJKD343	150	3549	1	5/1/2015	5/11/2015	2 Air	Calgary/Cape Town/Calgary, Kathmandu, Nepal	ASIA	MEAST	Diners			



Customer

Sales

Products

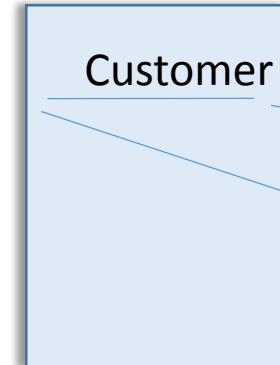
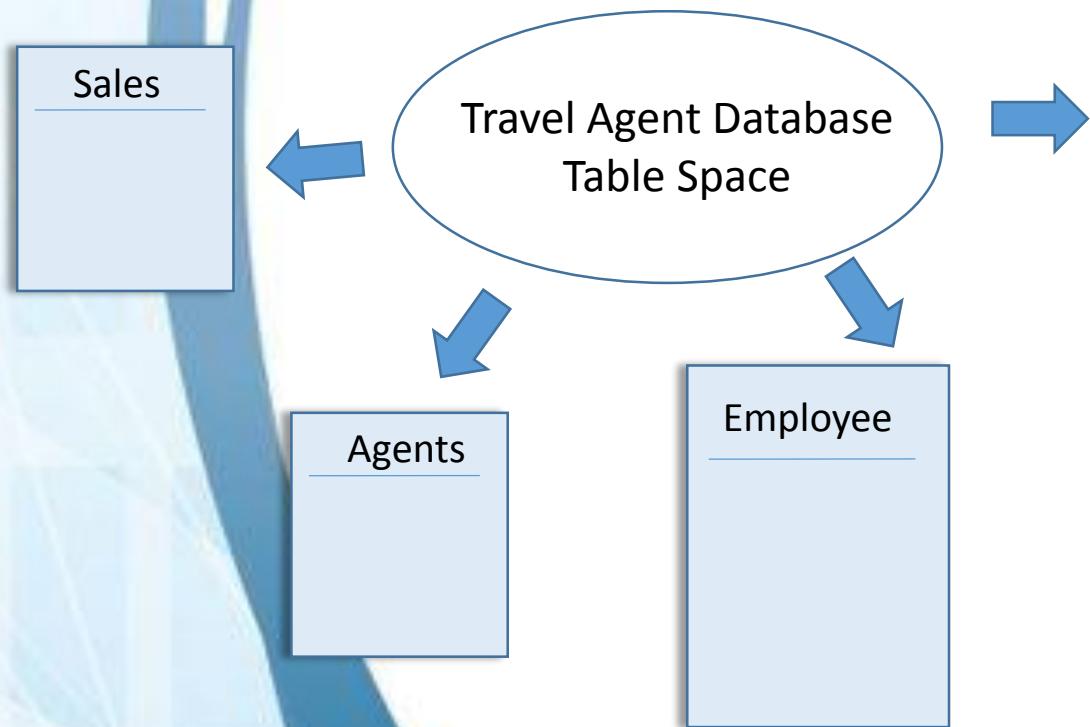
Suppliers



Excel → ORACLE®

Where we have been!

Your Data



Customer Table

Customer ID
Customer Name
Customer Phone Number
Customer Email Address
Customer Birthdate
Customer Significant Date
Customer Notes

CUSTOMER_ID	LAST_NAME	ADDRESS
214	Williams	80-204th Ave. #A3, SW
215	Chatman	9925 42nd Ave. #3B, SW
216	Dusenberry	6331 Durham Ave, SW
217	Citron	PO Box 1091, NW
218	Da Silva	90-36 53rd Avenue, SW
220	Segall	36 Brookdale Dr., SW
221	Malone	91-412 3rd Ave.1stFl, NW
223	Pace	13 Burlington Dr., NW
224	Diokno	44-206 4th St #6L, SE
225	Moffat	172 Lincoln St, NE
227	Heedles	932 Carnegie Ave., NE



USER MANAGEMENT PRIVILAGES, ROLES, MONITORING

Dec 2016

Where we have been!

Monitor Tables

Commissions

I'm a commission specialist, and only I can change this information



We are Agents, and we have access To view sales and product information

Sales

Products



I was trying to make unauthorized changes, and the log files caught me!

Suppliers



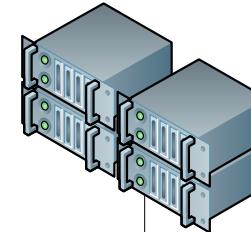
USERS

Suppliers

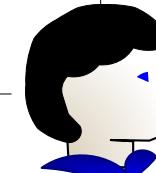
Products

Sales

Customer



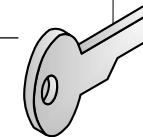
Oracle Server



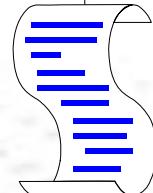
Users' Account



Roles



Privilege

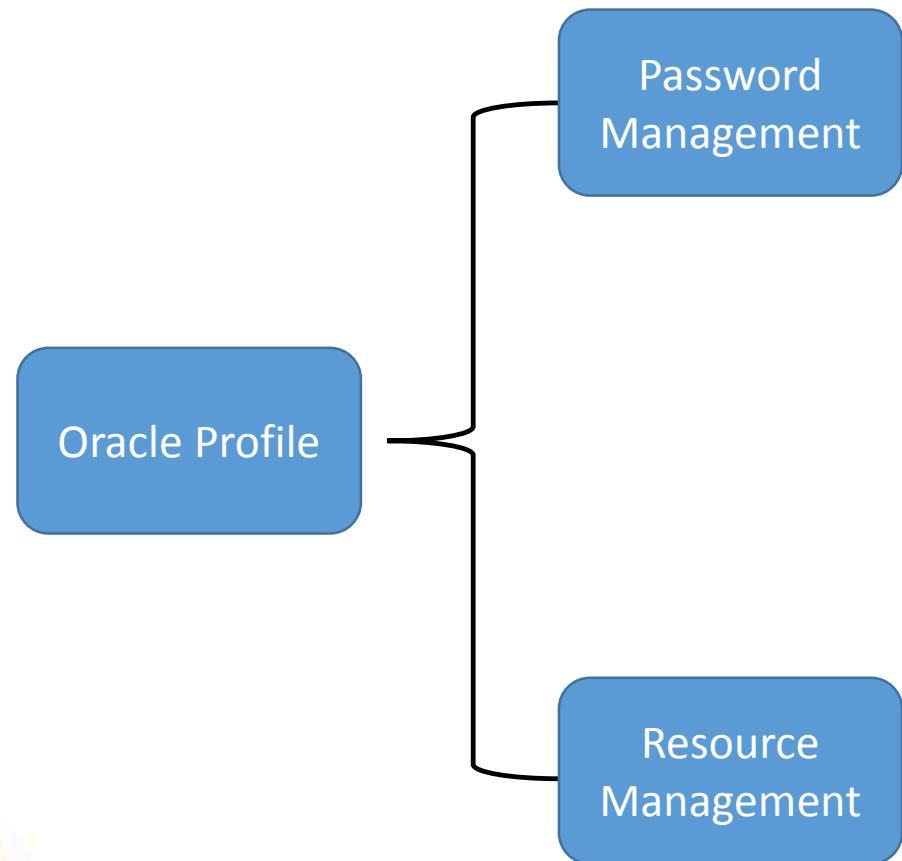


Profile

ROLES

User Level	Privilege
Intern	Create Session, select
Junior	Inter+ Insert/Update
Senior	Junior+ Delete except Employee, agent, commission
Manager	Full control on all tables
DBA	Full control of the database
Over watcher	Select on all tables

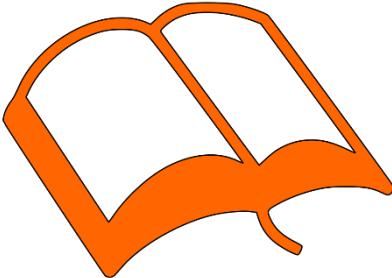
PROFILE



Password Management



User



Password History



Password Period



Oracle Server

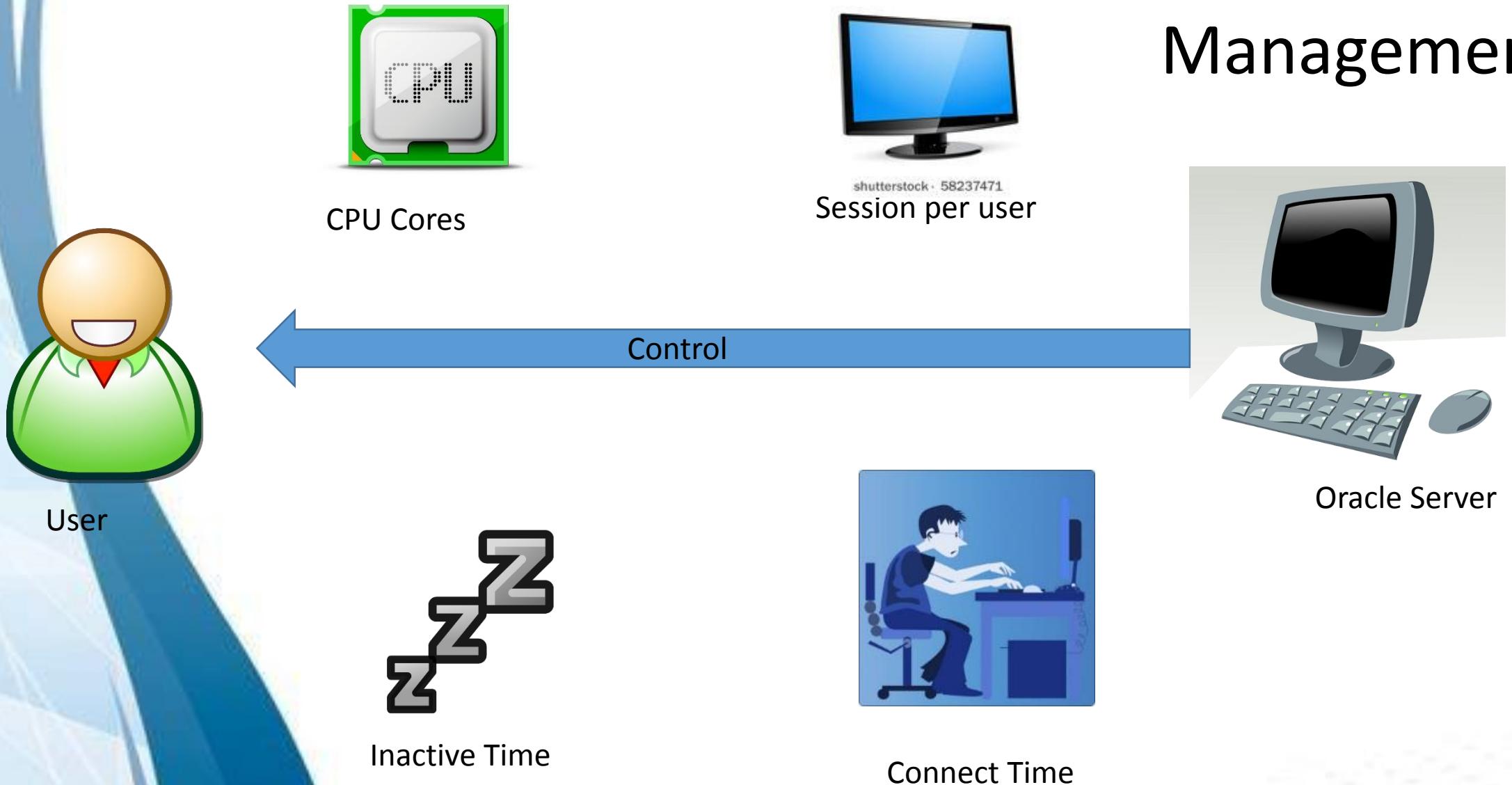


Account Lock



Password Verification

Resource Management

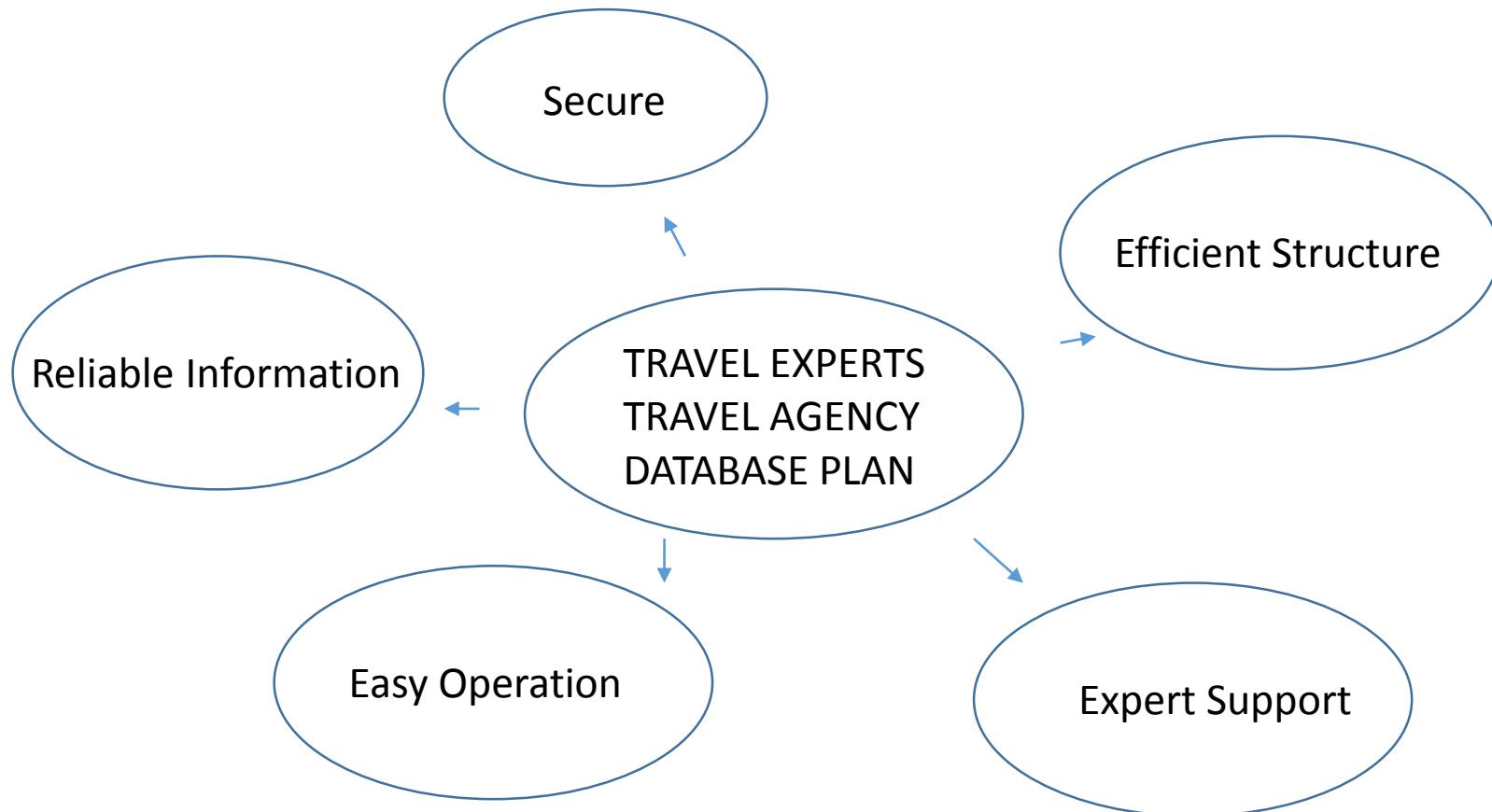




Where we have been!



Where we have been!



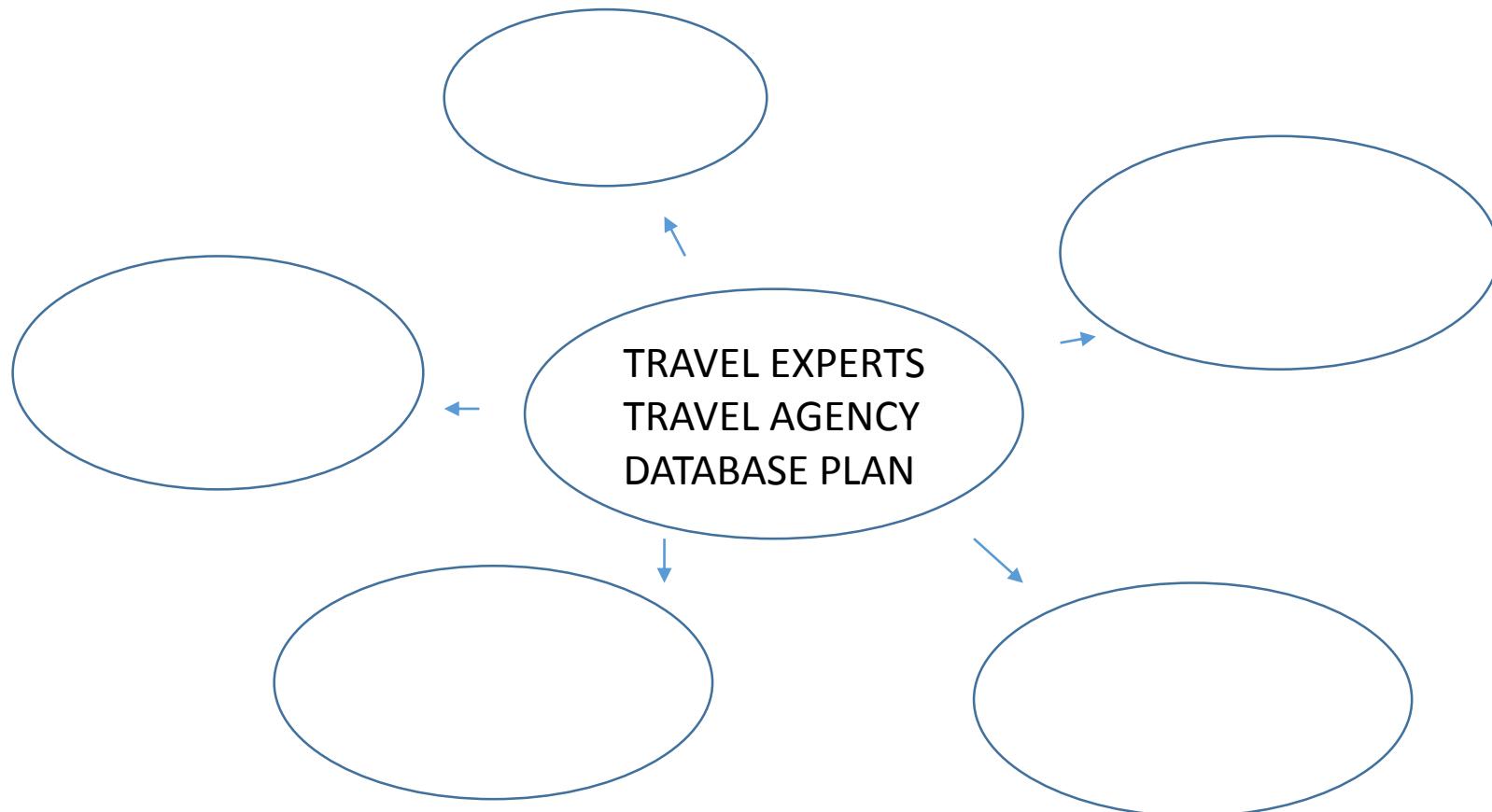


Where we will Go!



Landing on target!

Where we will go!





Elcaro2.0

Database Development

TRAVEL EXPERTS TRAVEL AGENCY





Project Team:

Mathew Ajani	– Lead Database Designer and Data Specialist
Kenny Vigar	– Data Recovery and Performance Lead
Fahim Azad	– Lead Data Modeler
Andy Gu	– Security and Permissions Expert

A combined total of 60 years experience!



Interview

© Randy Glasbergen
glasbergen.com



**"I want you to find a bold and innovative way
to do everything exactly the same way
it's been done for 25 years!"**

- How the agency works
- Issues we need to solve
- Business Requirements
- Budget
- Future upgrading

At the Beginning.....

Flat file

A	Value 1	Value 2	Value 3	Value 4	Value 5	Value 6	Value 7	Value 8	Value 9	Value 10
Row 1	91.5	14.04	18.88	77.82	68.03	8.08	98.18	20.33	45.55	41.53
Row 2	48.56	86.54	41.61	16	76.39	90.75	95.61	94.45	41.95	94.47
Row 3	96.34	25.1	7.37	67.03	93.7	69.75	23.15	82.51	31.1	96.27
Row 4	68.55	42.06	76.69	0.51	96.32	20.16	37.98	93.03	85.98	32.02
Row 5	61.37	85.77	33.14	63.03	90.1	95.77	18.26	57.61	18.44	72.12
Row 6	41.71	66.88	10.11	51.43	85.58	81.19	32.09	93	75.35	70.79
Row 7	34.27	30.49	28.46	93.24	30.43	49.31	41.5	20.36	73.19	18.11
Row 8	5.82	25.31	10.6	15.1	14.61	30.05	98.81	90.99	66.37	88
Row 9	4.93	41.99	73.61	58.95	45.08	88.01	24.59	86.86	58.53	87.81
Row 10	88.4	57.61	40.13	2.85	22.61	0.7	38.98	43.3	26.81	44.48
Row 11	79.85	78.27	88.04	15.89	39.22	44.66	3.57	4.16	53.95	67.08
Row 12	63.76	25.82	51.17	82.3	21.16	2.44	74.48	34.37	29.15	22.64
Row 13	39.94	92.54	26.4	17.78	57.01	52.96	42.77	37.06	57.26	8.54
Row 14	67.13	37.04	83.07	16.56	86.41	89.35	0.72	29.43	64.43	39.93
Row 15	29.81	69.17	54.43	99.75	41.38	84.12	96.4	39.08	57.98	75.59
Row 16	53.93	14.02	66.7	9.4	77.43	42.75	12.88	98.64	5.12	39.02
Row 17	7.96	99.91	23.67	8.73	46.51	82.28	86.92	34.78	84.09	73.23
Row 18	43.88	2.37	77.92	17.96	98.83	64.52	42.26	98.1	14.53	41.46
Row 19	63.36	14.56	23.96	14.12	31.01	12.63	23.43	76.03	39.73	25.86
Row 20	19.48	63.71	9.98	94.95	31.55	40.89	19.41	95.86	50.56	63.8

Microsoft Excel spread sheet

Sticky Notes



Issues



- Commissions tracking and calculation
- Invoice generating
- Errors in the spread sheet
- Missing data in the spread sheet
- Data duplicity in the spread sheet
- Data secure issue

What we did...

Develop a form to enter new data into the database

Need to load all existing data into the database

Create set of scripts or program, which can be easily modified to produce monthly reports

Correct the data into database

Secure all the data

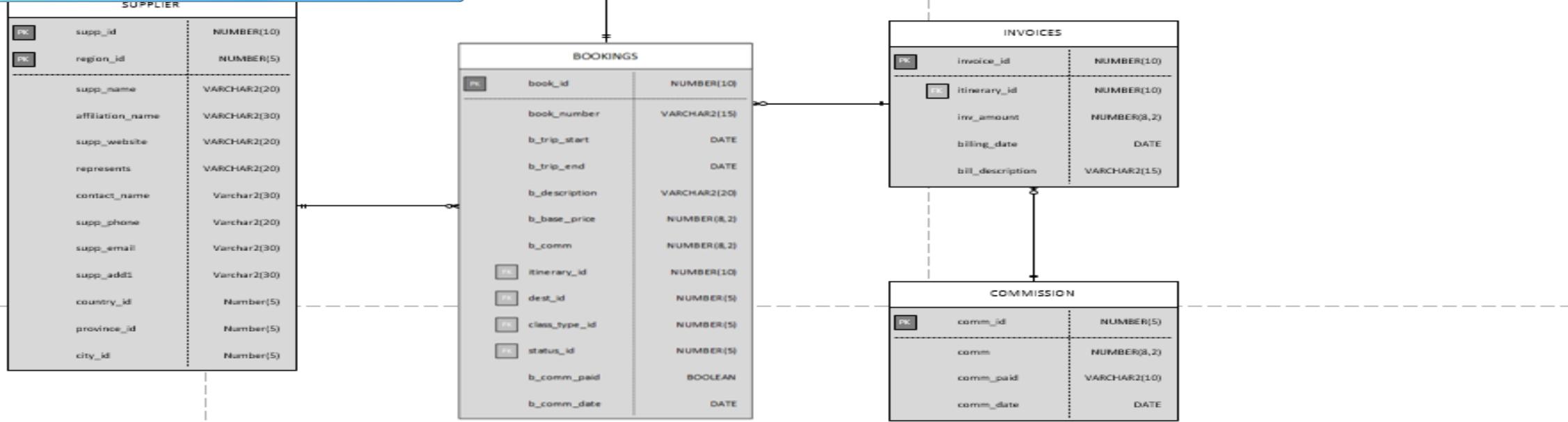
Design easy to use graphical interface

Make sure database work throughout the local network

How we did

Excel sheet

	A	B	C	D	E	F	G	H	I	J	K
1	Value 1	Value 2	Value 3	Value 4	Value 5	Value 6	Value 7	Value 8	Value 9	Value 10	
2	Row 1	91.5	14.04	18.88	77.82	68.03	8.08	98.18	20.33	45.55	41.53
3	Row 2	48.56	86.54	41.61	16	76.39	90.75	95.61	94.45	41.95	94.47
4	Row 3	96.34	25.1	7.37	67.03	93.7	69.75	23.15	82.51	31.1	96.27
5	Row 4	68.55	42.06	76.69	0.51	96.32	20.16	37.98	93.03	85.98	32.02
6	Row 5	61.37	85.77	33.14	63.03	90.1	95.77	18.26	57.61	18.44	72.12
7	Row 6	41.71	66.88	10.11	51.43	85.58	81.19	32.09	93	75.35	70.79
8	Row 7	34.27	30.49	28.46	93.24	30.43	49.31	41.5	20.36	73.19	18.11
9	Row 8	5.82	25.31	10.6	15.1	14.61	30.05	98.81	90.99	66.37	88
10	Row 9	4.93	41.96	73.61	58.95	45.08	88.01	24.59	86.86	58.53	87.81
11	Row 10	88.4	57.61	40.13	2.85	22.61	0.7	38.98	43.3	26.81	44.48
12	Row 11	79.85	78.27	88.04	15.89	39.22	44.66	3.57	4.16	53.95	67.08
13	Row 12	63.76	25.82	51.17	82.3	21.16	2.44	74.48	34.37	29.15	22.64
14	Row 13	39.94	92.54	26.4	17.78	57.01	52.96	42.77	37.06	57.26	8.54
15	Row 14	67.13	37.04	83.07	16.56	86.41	89.35	0.72	29.43	64.43	39.93
16	Row 15	29.81	69.17	54.43	99.75	41.38	84.12	96.4	39.08	57.98	75.59
17	Row 16	53.93	14.02	66.7	9.4	77.43	42.75	12.88	98.64	5.12	39.02
18	Row 17	7.96	99.91	23.67	8.73	46.51	82.28	86.92	34.78	84.09	73.23
19	Row 18	43.88	2.37	77.92	17.96	98.83	64.52	42.26	98.1	14.53	41.46
20	Row 19	63.36	14.56	23.96	14.12	31.01	12.63	23.43	76.03	39.73	25.86
21	Row 20	19.48	63.71	9.98	94.95	31.55	40.89	19.41	95.86	50.56	63.8

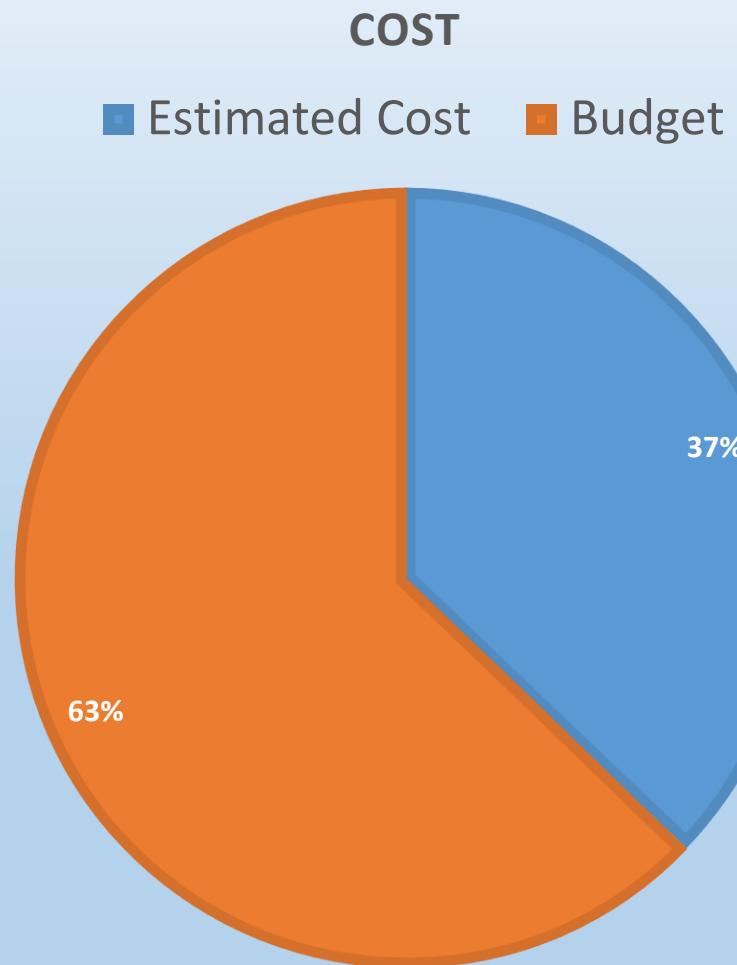


Future Update

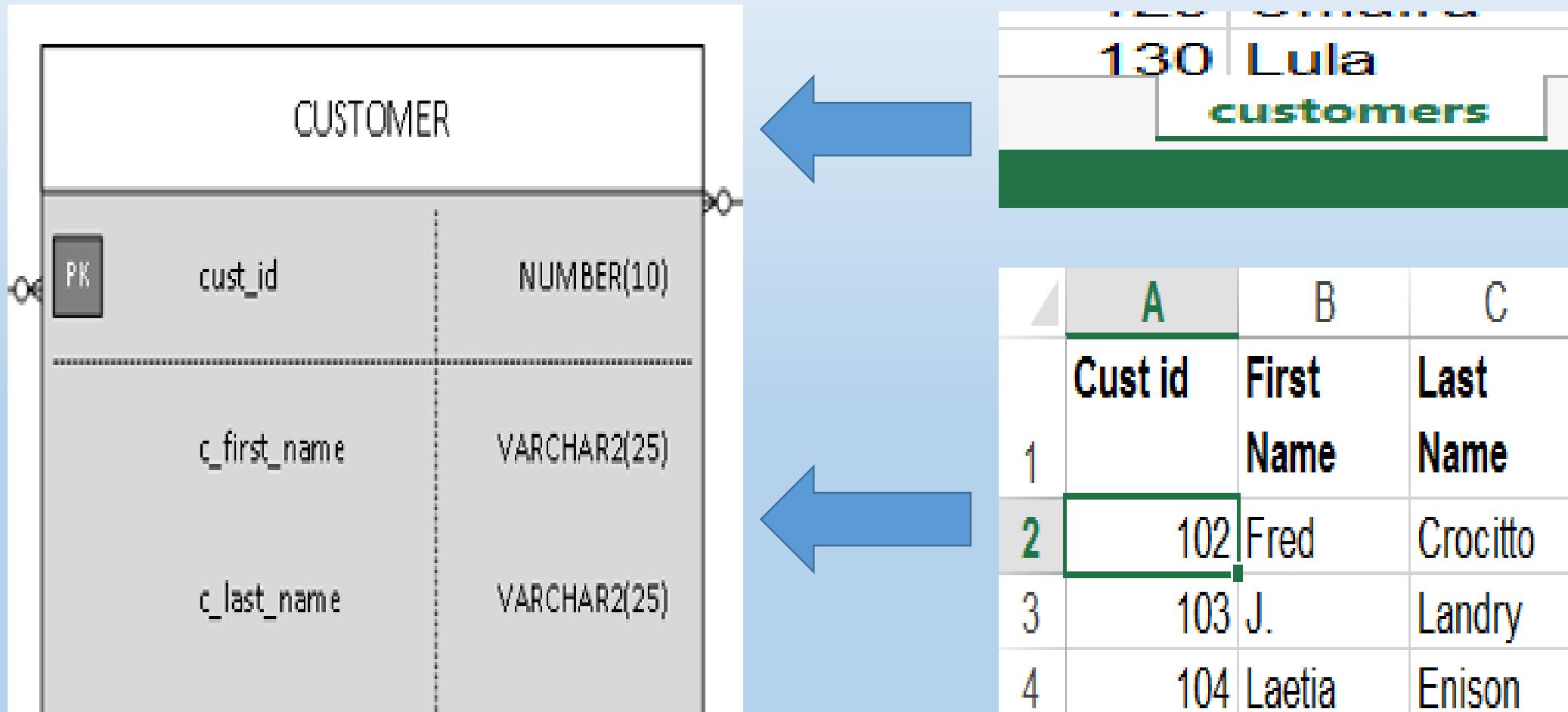
Website development

Cloud function

Cost

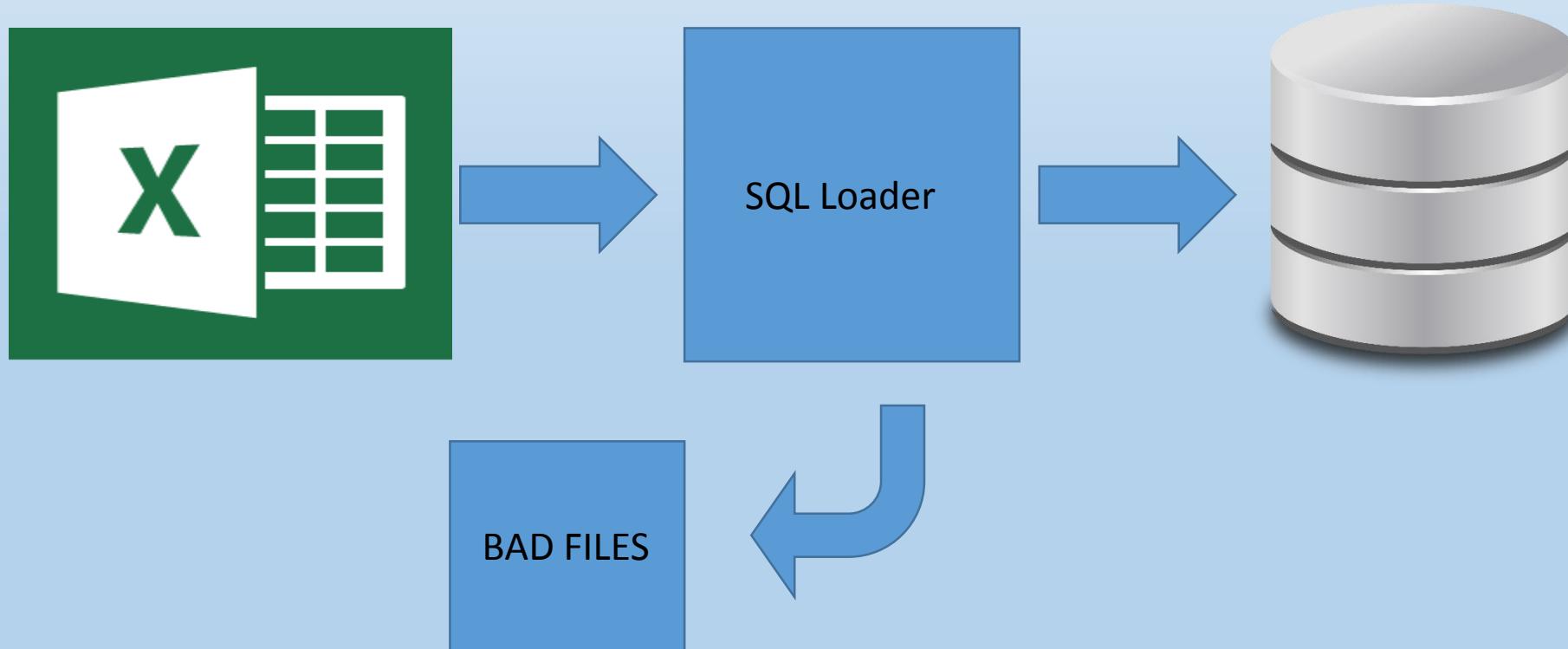


How We Created Tables



How We Loaded Data

SQL Loader (Data loading application)



Creating Users and Roles

Role are assigned privileges

Roles are granted to Users Depending on their Responsibilities

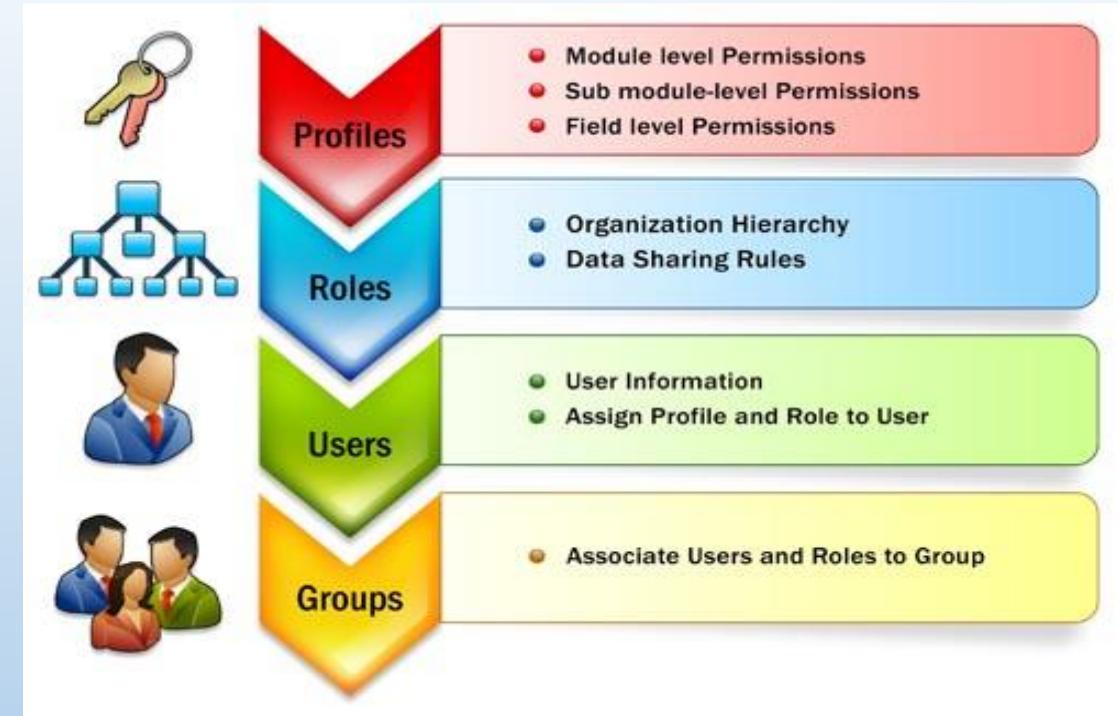
A User can be granted more than one role

Example:

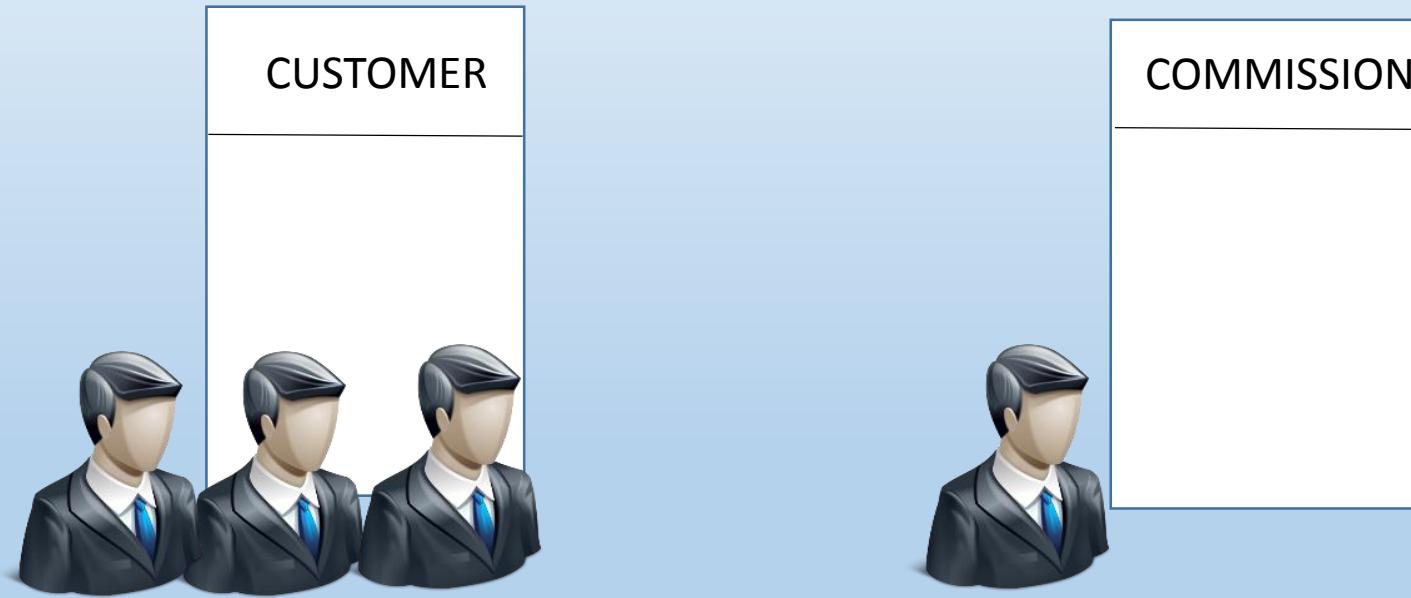
Role - Junior Agent

Privileges - Create table, select, insert etc

User – Bob is granted Junior Agent



Creating Users and Roles



Different Users have Different Access Level

Creating User Profiles

Limit for Resource Allocation and Usage

Example: Session Per User

CPU Per Session

Idle Time

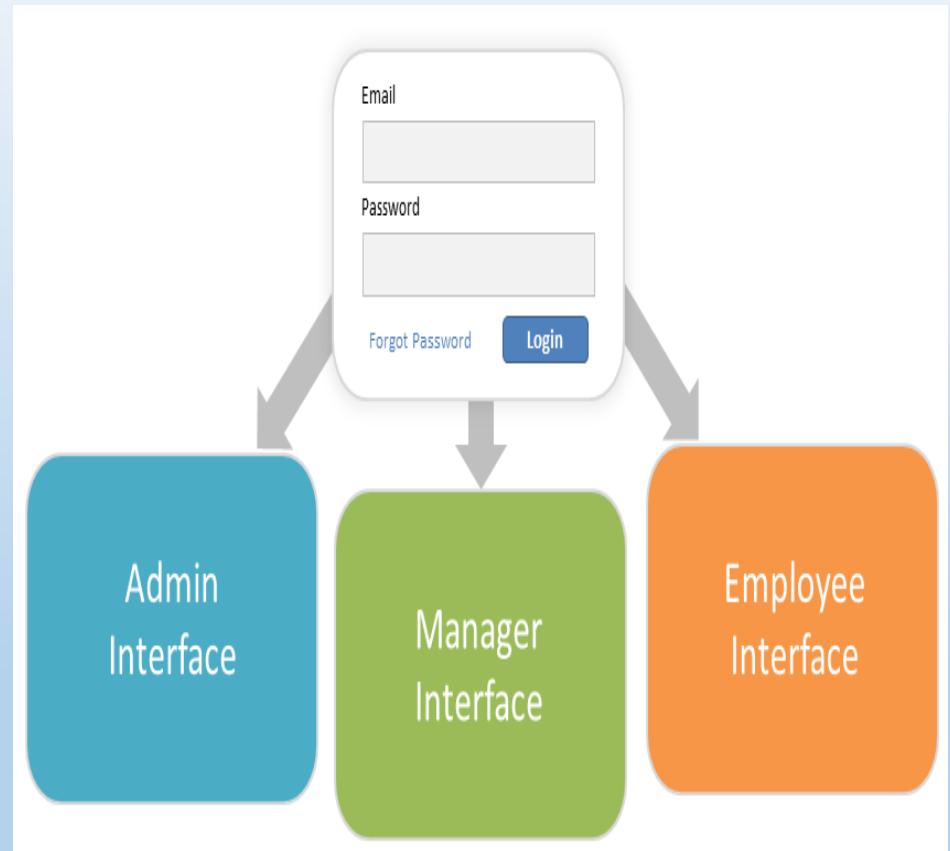
CPU Per Call

System and User Security

Example: Failed Login Attempt

Password Lock Time

Password life time

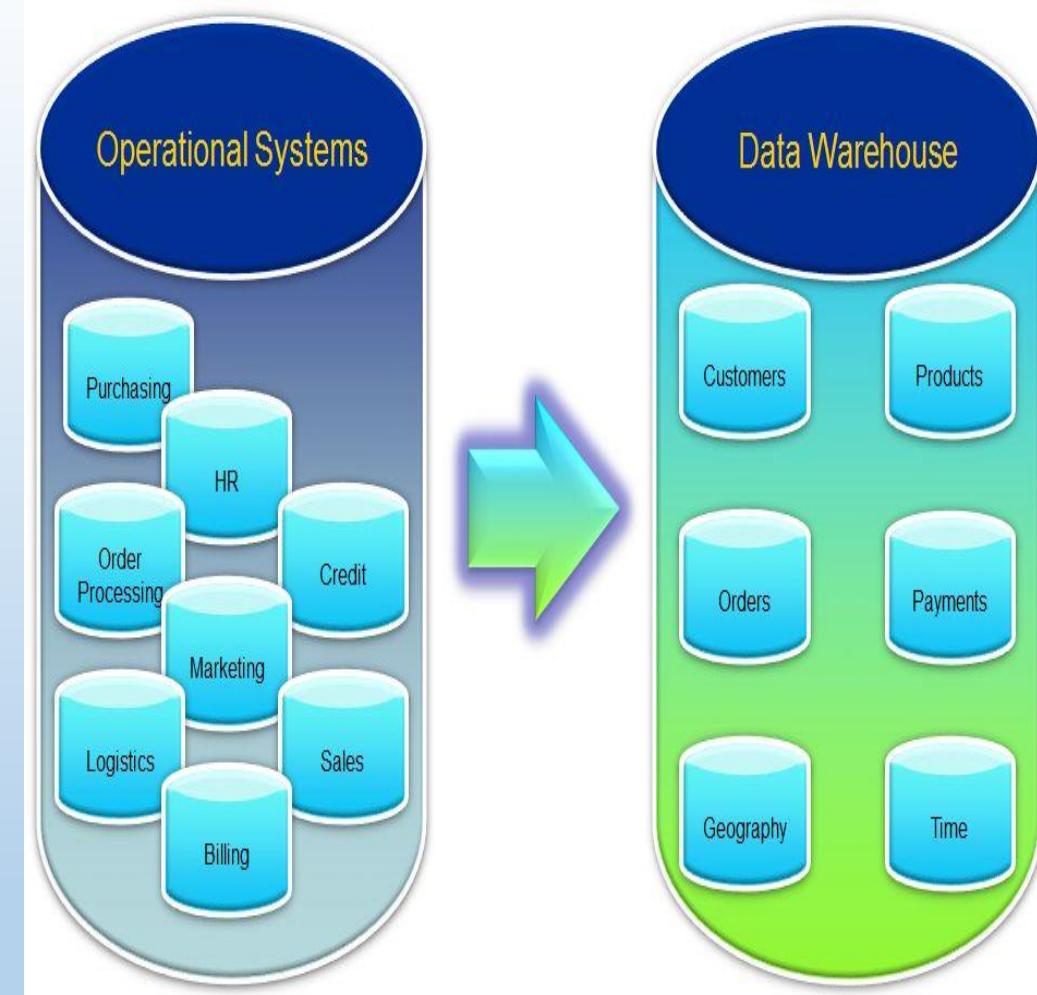


Data Warehouse

Data Storage for Historical Reports and Analysis

Can be used to Determine Patterns and to Make Predictions

Current and Historical Data are Stored in one Single Place

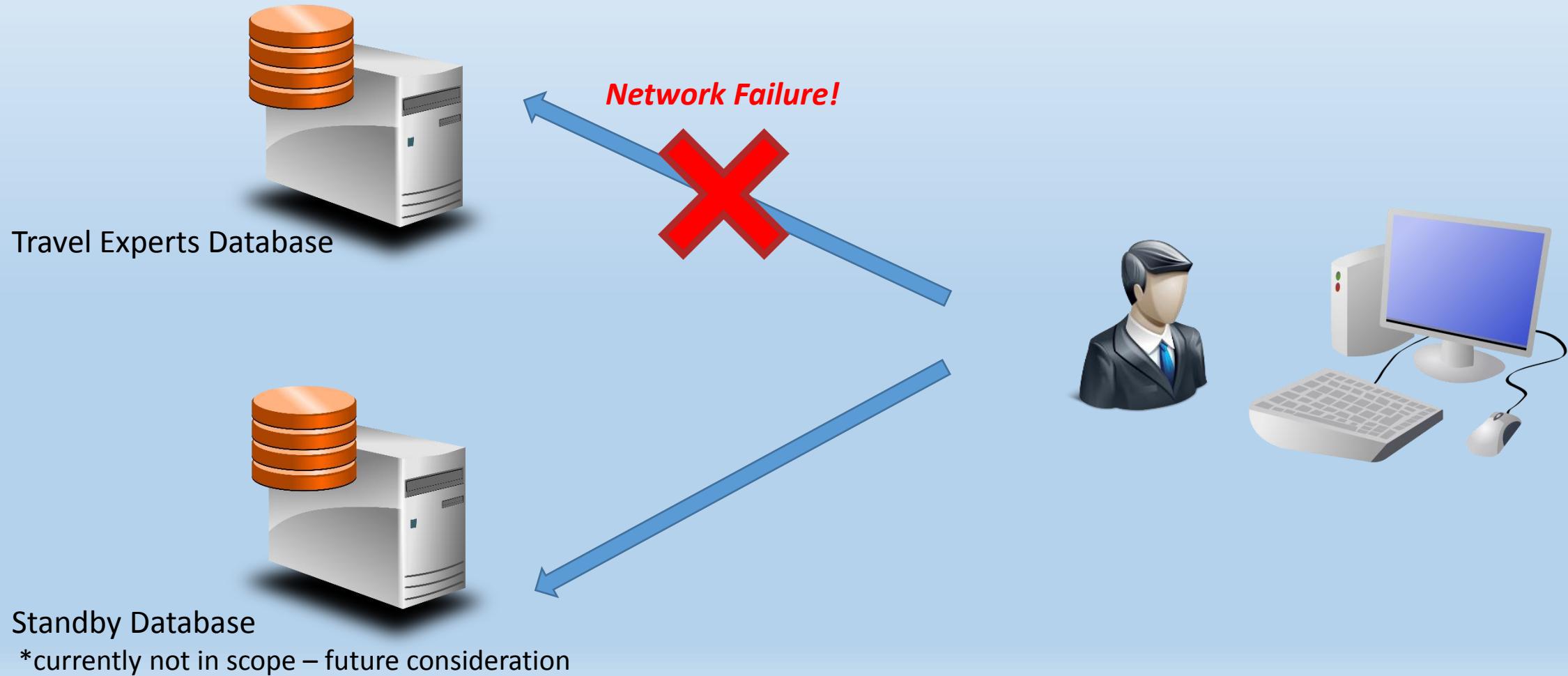


Data Warehouse

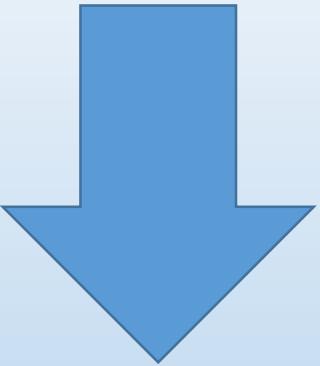
Performance and Tuning



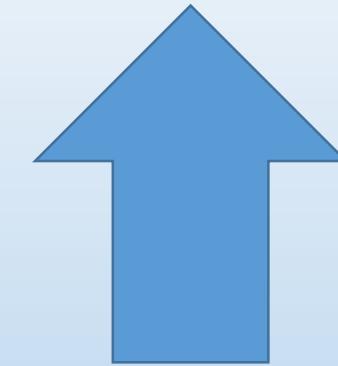
High Availability = Normal functionality even with Hardware or Network Failures
= Faster Disaster Recovery and No Data Loss
= Increased Productivity



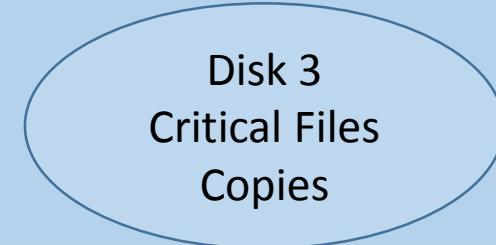
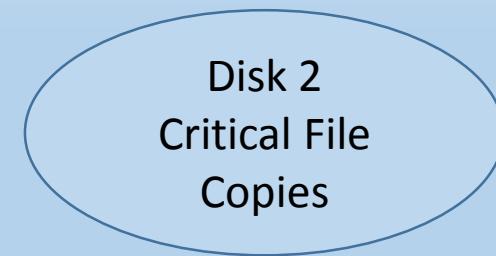
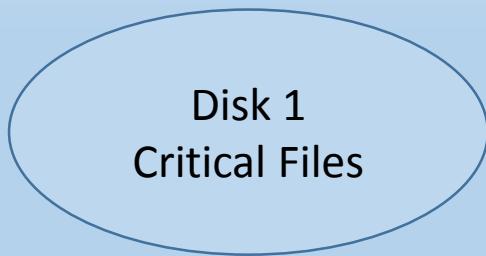
Backup and Recovery



Decrease the Mean Time To Recovery
Fast response for data loss or errors



Increase the Mean Time Between Failure
Keep issues from occurring regularly or at all!



Multiplex Files Over Multiple Disks

Backup and Recovery

- Backups preformed nightly
- “incremental” for speed of backup
- All critical database files as well as all user data entered

Most Importantly

- Time is scheduled to verify backups work, and practice backup strategy.

*What good is your backup if you don't know if it will work?

Backups kept off site in case of emergency



Performance Tuning



Enhance your Database Performance by

- Making sure your server hardware is adequate
- Keep testing scripts to your Test database – Leave Production environments stable
- Verify instance parameters regarding Memory
- Use ORACLE tools such as Enterprise Manager to analyze SQL Code – Can it be improved?
- Build performance based indexes on frequently used tables (Commission Reporting).

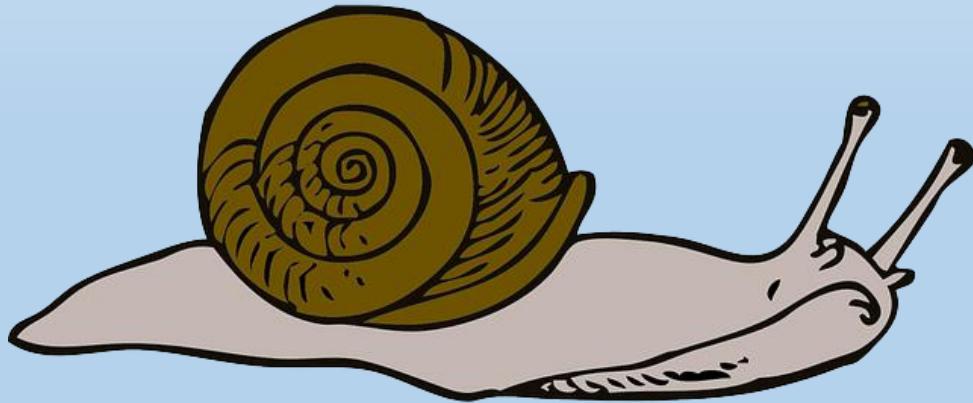


Your Database Search = Your Execution Plan

Execution Plan

Elapsed: 00:00:00.17
4 Cost (%CPU)

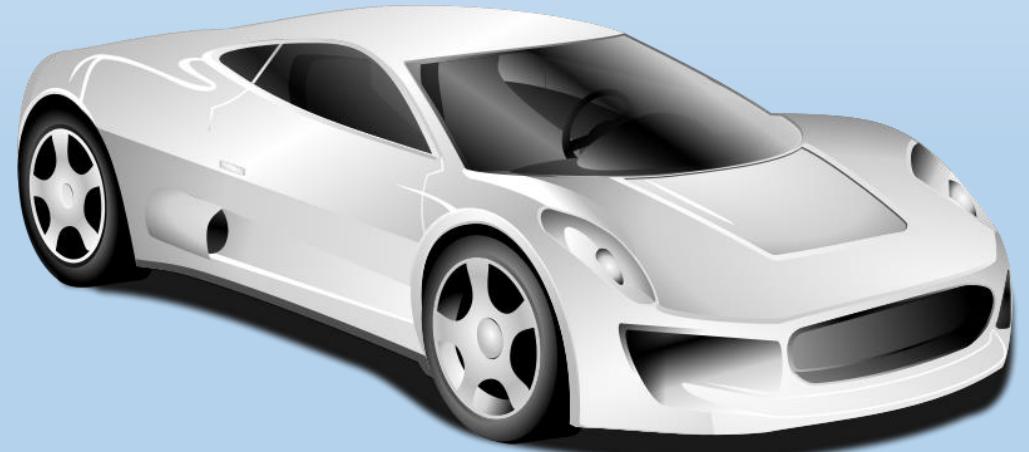
TABLE ACCESS FULL



Execution Plan

Elapsed: 00:00:00.06 Cost (%CPU)
3 Cost (%CPU)

ACCESS BY INDEX

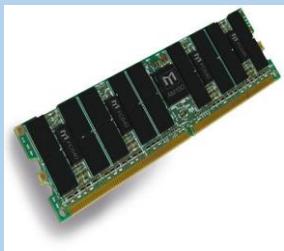


Server and Database

Configuration



XEON E3-1200
4 Cores Hyper Threading



32 GB DDR4 ECC



CENTOS 7
KERNEL 3.16



Oracle 12c Standard

Why Linux?



Stability



Security

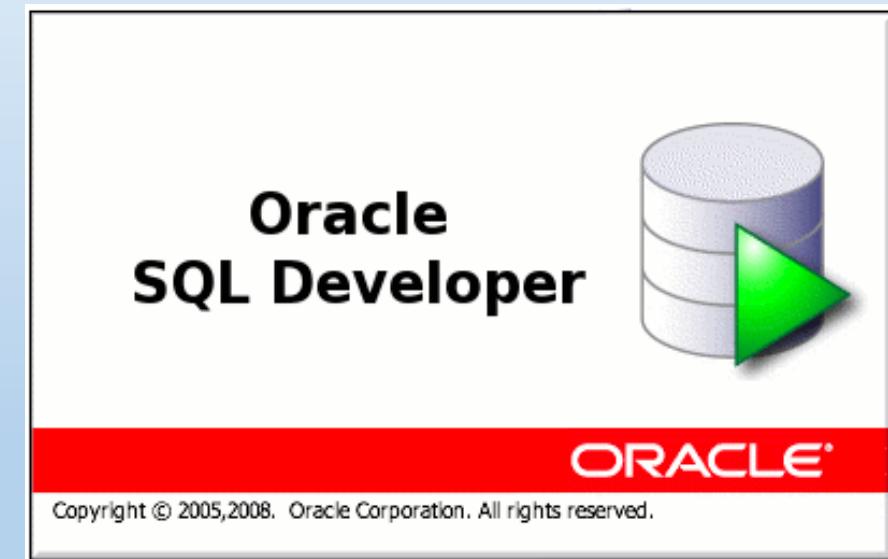


Total Cost of Ownership

Application and Scripts



ORACLE®
Application Express

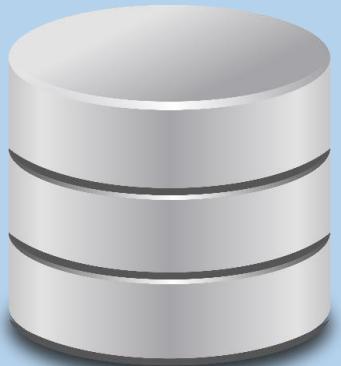


THANK YOU



Elcaro2.0
Database Development

TRAVEL EXPERTS TRAVEL AGENCY

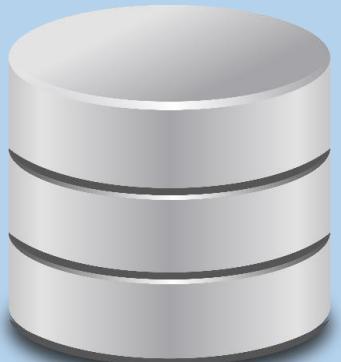


QUESTIONS?



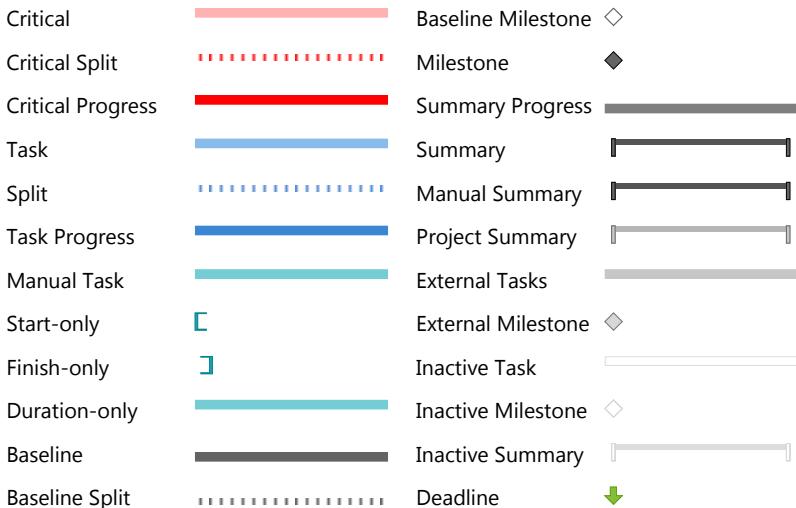
Elcaro2.0
Database Development

TRAVEL EXPERTS TRAVEL AGENCY



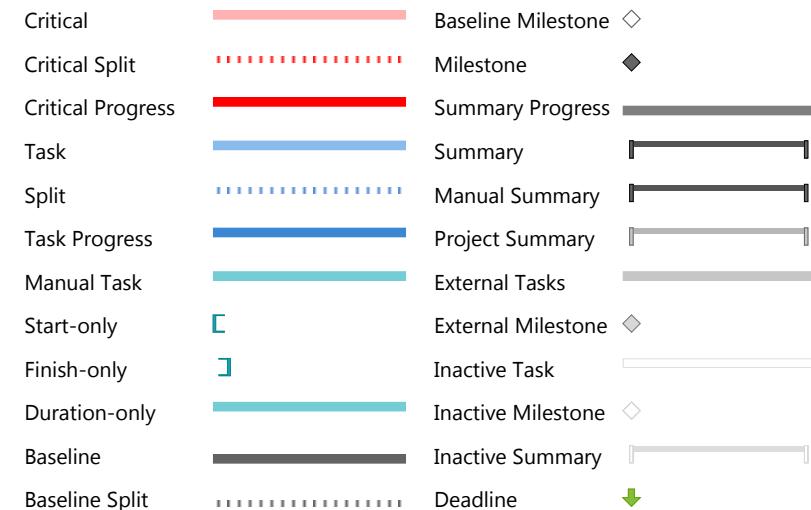
Kenny Vigar - DBA Schedule.mpp

ID	Task Mode	Task Name	Duration	Start
1	★	Entire Project Timeline	73 days	Tue 10/25/16
2	☛	Analysis	7 days	Tue 10/25/16
3	☛	Select Team and Project Mgr	1 day	Tue 10/25/16
4	☛	Define the Project Plan (Informational Meetings)	1 day	Wed 10/26/16
5	☛	Define Business Requirements	1 day	Thu 10/27/16
6	☛	Define Budget Plan	1 day	Fri 10/28/16
7	☛	Initial Data Cleanup	3 days	Sat 10/29/16
8	☛	Procure Hardware	1 day	Thu 11/3/16
9	☛	Puchase Server/Monitors/New Laptops	1 day	Thu 11/3/16
10	☛	Design Database	11 days	Fri 11/4/16
11	☛	Create ERD - Entities, Attributes and Relationships	7 days	Fri 11/4/16
12	☛	BiWeekly Meeting with Stakeholders	4 days	Tue 11/15/16
13	☛	Normalization	4 days	Tue 11/15/16
14	★	Develop Database	20 days	Mon 11/21/16
15	☛	Create Database Blueprint	1 day	Mon 11/21/16
16	☛	Create Pfile	2 days	Tue 11/22/16
17	☛	Create Database Script	2 days	Thu 11/24/16
18	☛	Create Tablespace Script	2 days	Fri 11/25/16
19	☛	BiWeekly Meeting with Stakeholders	0 days	Tue 11/29/16
20	☛	Create Database Load Data	20 days	Tue 11/29/16
21	☛	Further Data Cleanup	3 days	Tue 12/27/16
22	☛	Table Creation	4 days	Fri 12/30/16
23	☛	Load Data	5 days	Thu 1/5/17
24	☛	Test Database Table Relations	5 days	Thu 1/12/17
25	☛	Create Users and Privileges	2 days	Wed 1/18/17
26	☛	Confirm User Permissions with TEA	1 day	Fri 1/20/17
27	☛	Create User Profiles and Test Limits	2 days	Mon 1/23/17
28	☛	Create User Roles	2 days	Wed 1/25/17



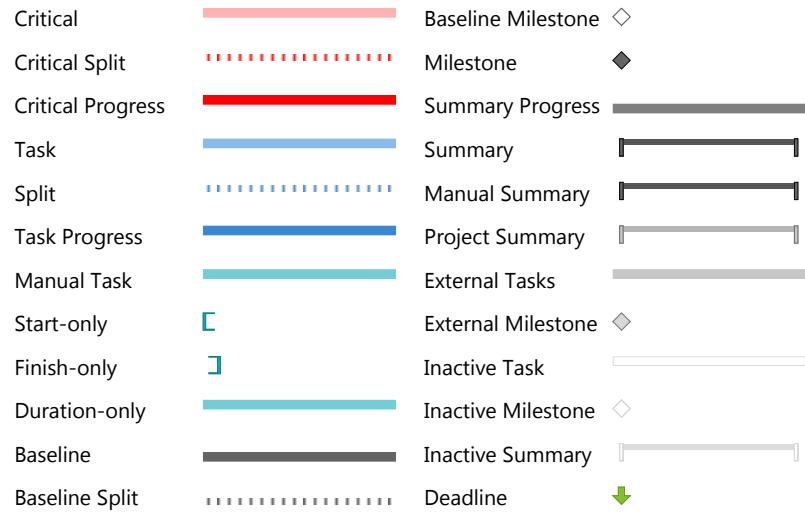
Kenny Vigar - DBA Schedule.mpp

ID	Task Mode	Task Name	Duration	Start
29	█	Create User Accounts with Expired Pwds and Lock	1 day	Fri 1/27/17
30	█	Setup Network Access to Database	3 days	Fri 1/27/17
31	█	Planning TNS information (gather computer host addresses)	1 day	Tue 1/31/17
32	█	Setup Tracing/Logging	3 days	Tue 1/31/17
33	█	Setup Server LISTENER.ORA	1 day	Thu 2/2/17
34	█	Setup Client machines TNSNAMES.ORA and SQLNET.ORA	1 day	Fri 2/3/17
35	█	Extensive Network Access Testing	2 days	Sat 2/4/17
36	█	Design GUI Interface	10 days	Wed 2/8/17
37	█	Design User Interface for Entering Data (GUI)	10 days	Wed 2/22/17
38	█	BiWeekly Meeting with Stakeholders	0 days	Tue 3/7/17
39	█	Database Testing	3 days	Wed 3/8/17
40	█	Test DB functionality, test reports and queries	3 days	Wed 3/8/17
41	█	Database Security	3 days	Mon 3/13/17
42	█	Create user profiles, roles, and assign permissions	2 days	Mon 3/13/17
43	█	Create scripts for owner/manager to add new users	1 day	Wed 3/15/17
44	█	Develop Front End GUI	7 days	Thu 3/16/17
45	█	Design GUI interface to load new data	6 days	Thu 3/16/17
46	█	Test process of importing data with GUI	1 day	Fri 3/24/17
47	█	BiWeekly Meeting with Stakeholders	0 days	Fri 3/24/17
48	█	Hardware Setup	6 days	Mon 3/27/17
49	█	Deploy new user computers	2 days	Mon 3/27/17
50	█	Deploy database server	2 days	Wed 3/29/17
51	█	Verify current network infrastructure	2 days	Fri 3/31/17
52	█	Load any further data/Backup Data	6 days	Tue 4/4/17
53	█	Load any additional data	2 days	Tue 4/4/17
54	█	Preform initial backup of data	2 days	Thu 4/6/17
55	█	Preform test data restoration	2 days	Mon 4/10/17



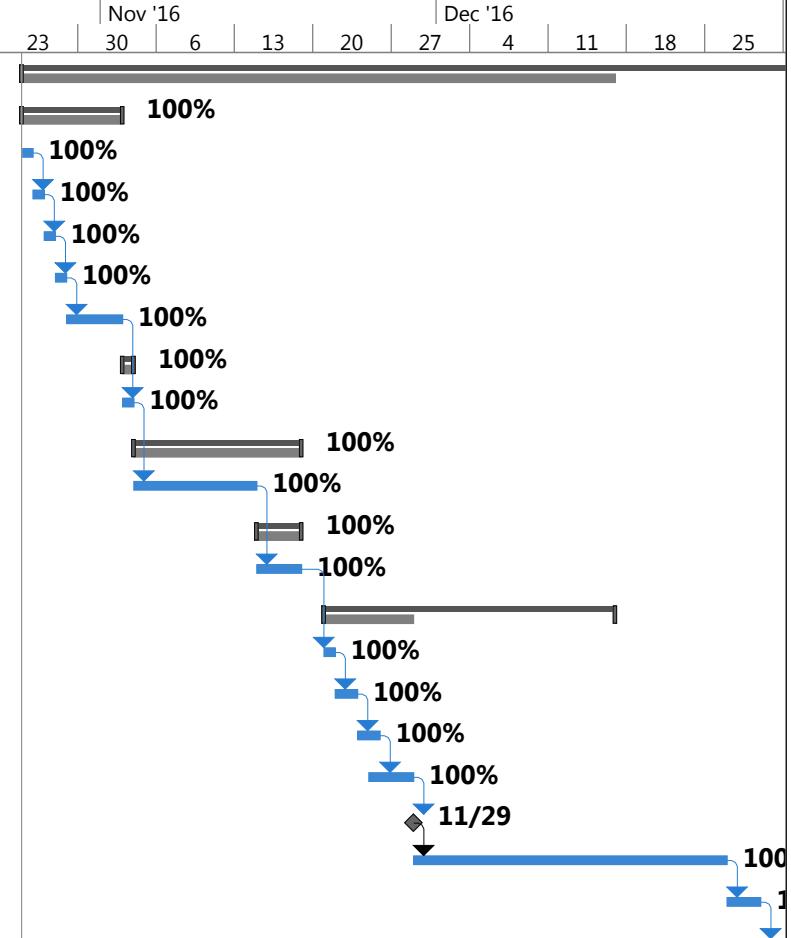
Kenny Vigar - DBA Schedule.mpp

ID	Task Mode	Task Name	Duration	Start
56		BiWeekly Meeting with Stakeholders	0 days	Tue 4/11/17
57		Rollout	1 day	Wed 4/12/17

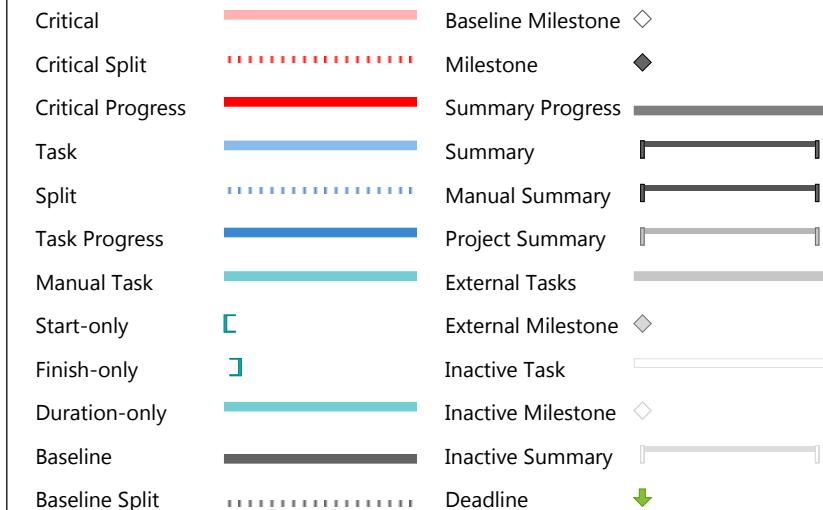


Kenny Vigar - DBA Schedule.mpp

Finish	% Complete	Nov '16	Dec '16
Thu 2/2/17	100%		
Wed 11/2/16	100%		
Tue 10/25/16	100%		
Wed 10/26/16	100%		
Thu 10/27/16	100%		
Fri 10/28/16	100%		
Wed 11/2/16	100%		
Thu 11/3/16	100%		
Thu 11/3/16	100%		
Fri 11/18/16	100%		
Mon 11/14/16	100%		
Fri 11/18/16	100%		
Fri 11/18/16	100%		
Fri 12/16/16	100%		
Mon 11/21/16	100%		
Wed 11/23/16	100%		
Fri 11/25/16	100%		
Mon 11/28/16	100%		
Tue 11/29/16	100%		
Mon 12/26/16	100%		
Thu 12/29/16	100%		
Wed 1/4/17	100%		
Wed 1/11/17	100%		
Wed 1/18/17	100%		
Thu 1/19/17	100%		
Fri 1/20/17	100%		
Tue 1/24/17	100%		
Thu 1/26/17	100%		

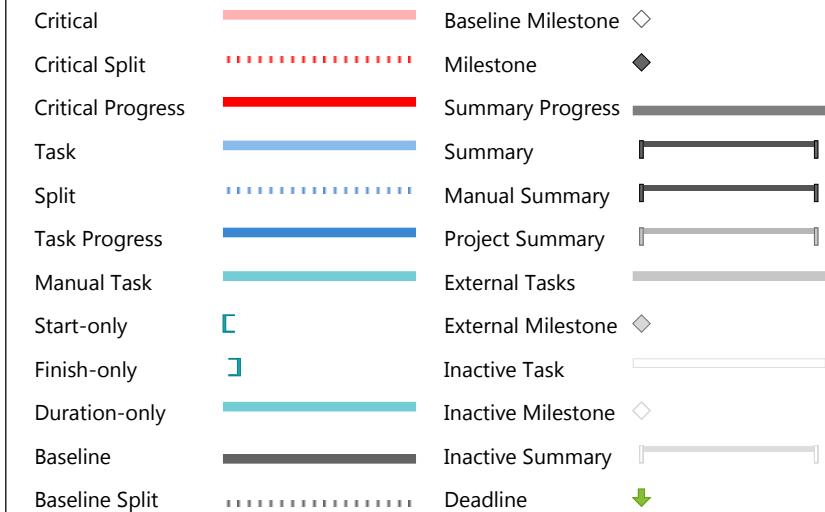


100% 11/29



Kenny Vigar - DBA Schedule.mpp

Finish	% Complete		Nov '16		Dec '16							
		16	23	30	6	13	20	27	4	11	18	25
Fri 1/27/17	100%											
Tue 1/31/17	100%											
Tue 1/31/17	100%											
Thu 2/2/17	100%											
Thu 2/2/17	100%											
Fri 2/3/17	100%											
Tue 2/7/17	100%											
Tue 2/21/17	100%											
Tue 3/7/17	100%											
Tue 3/7/17	100%											
Fri 3/10/17	100%											
Fri 3/10/17	100%											
Wed 3/15/17	100%											
Tue 3/14/17	100%											
Wed 3/15/17	100%											
Fri 3/24/17	100%											
Thu 3/23/17	100%											
Fri 3/24/17	100%											
Fri 3/24/17	100%											
Mon 4/3/17	100%											
Tue 3/28/17	100%											
Thu 3/30/17	100%											
Mon 4/3/17	100%											
Tue 4/11/17	100%											
Wed 4/5/17	100%											
Fri 4/7/17	100%											
Tue 4/11/17	100%											

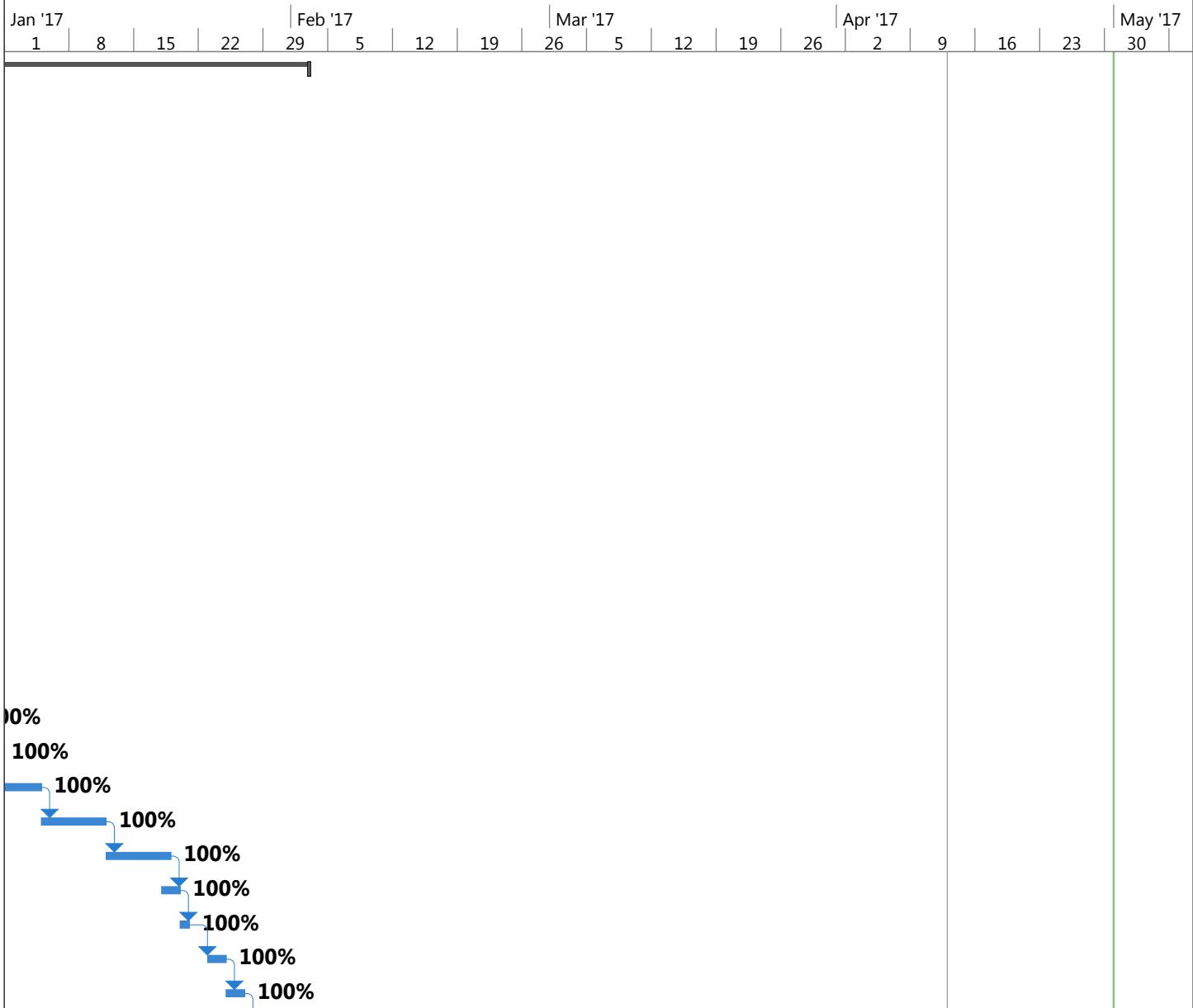


Kenny Vigar - DBA Schedule.mpp

Finish	% Complete		Nov '16		Dec '16	
Tue 4/11/17	100%		16 23 30 Nov '16 6 13 20 27 4 11 18 25			
Wed 4/12/17	100%					

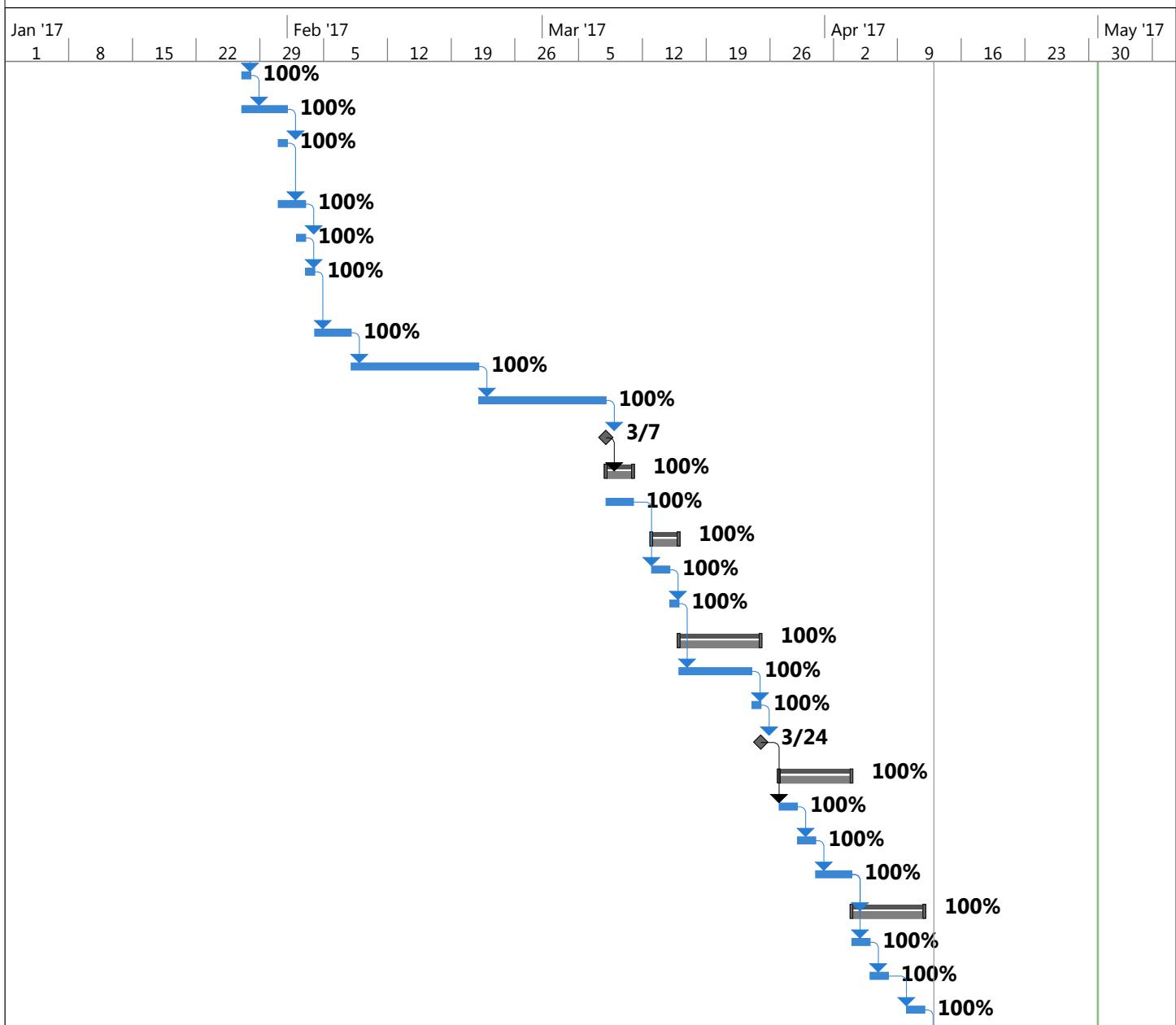
Critical		Baseline Milestone	
Critical Split		Milestone	
Critical Progress		Summary Progress	
Task		Summary	
Split		Manual Summary	
Task Progress		Project Summary	
Manual Task		External Tasks	
Start-only		External Milestone	
Finish-only		Inactive Task	
Duration-only		Inactive Milestone	
Baseline		Inactive Summary	
Baseline Split		Deadline	

Kenny Vigar - DBA Schedule.mpp



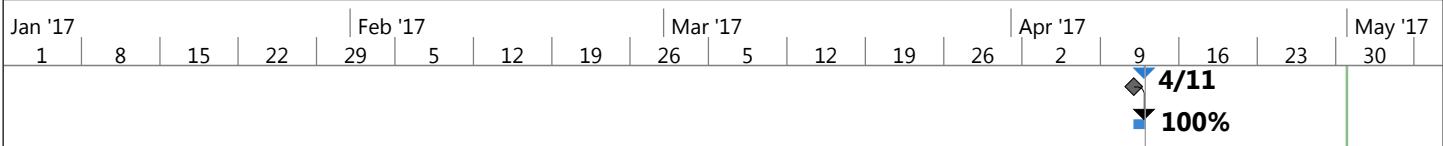
Critical		Baseline Milestone	
Critical Split		Milestone	
Critical Progress		Summary Progress	
Task		Summary	
Split		Manual Summary	
Task Progress		Project Summary	
Manual Task		External Tasks	
Start-only		External Milestone	
Finish-only		Inactive Task	
Duration-only		Inactive Milestone	
Baseline		Inactive Summary	
Baseline Split		Deadline	

Kenny Vigar - DBA Schedule.mpp



Critical		Baseline Milestone	
Critical Split		Milestone	
Critical Progress		Summary Progress	
Task		Summary	
Split		Manual Summary	
Task Progress		Project Summary	
Manual Task		External Tasks	
Start-only		External Milestone	
Finish-only		Inactive Task	
Duration-only		Inactive Milestone	
Baseline		Inactive Summary	
Baseline Split		Deadline	

Kenny Vigar - DBA Schedule.mpp



Critical		Baseline Milestone	
Critical Split		Milestone	
Critical Progress		Summary Progress	
Task		Summary	
Split		Manual Summary	
Task Progress		Project Summary	
Manual Task		External Tasks	
Start-only		External Milestone	
Finish-only		Inactive Task	
Duration-only		Inactive Milestone	
Baseline		Inactive Summary	
Baseline Split		Deadline	