MONITORING OF GROWTH MANAGEMENT TRENDS USING TIDEMARK/ADVANTAGE

June 2005

Prepared by:

CLARK COUNTY, WASHINGTON

and

ANGELO EATON & Associates

This project was made possible through the cooperation of each city and through a grant from the State of Washington Department of Community Trade and Economic Development.

TABLE OF CONTENTS

INTRODUCTION

Under the GMA, comprehensive plans and development regulations are subject to "continuing review and evaluation." In addition to the general requirements for review described in RCW 36.70A.130, RCW 36.70A.215 established a special review and evaluation program requirement for the six largest counties in Western Washington, referred to as the Buildable Lands Program. Implementation of this program is required for Clark County and cities within its boundaries. In accordance with the Buildable Lands Program, Clark County must monitor the intensity and density of development to determine whether the county and its cities are achieving sufficient urban densities within urban growth areas by comparing growth and development assumptions, targets, and objectives contained in the county-wide planning policies and the county and city comprehensive plans with actual growth and development that has occurred in the county and its cities.

The base data needed for the Buildable Lands Program must enable Clark County to determine:

- Whether there is sufficient suitable land to accommodate the county-wide population projection and the subsequent population allocations within Clark County and between the county and its cities.
- The actual density of housing that has been constructed and the actual amount of land developed for commercial and industrial uses within the urban growth area.

The process also allows for testing assumptions used by the county and its cities in the Vacant Lands Analysis for calculating land needed to establish urban growth boundaries and achieve growth objectives.

Clark County has been collecting buildable lands data since the early 1990's. The process has been time consuming for county staff because the data generated from each community must be assembled into one master spreadsheet. It has been the county's practice to accept the data in whatever form the cities will provide. County staff must reconcile differences in the data collected, reconcile errors, and sometimes perform the raw data collection. Buildable lands data collection can be time consuming for city staff because they must compile the data and forward it to the county. The data is primarily generated from building permit records. In some communities, building officials are so busy that it is difficult to make collection of data in a format useful to planners a priority. The CTED grant provided a valuable opportunity to unify buildable lands data into one system and to make collection and analysis easier for individual cities and the county.

The grant also provided an opportunity to use a powerful permit tracking system for a land use planning purpose. Counties and cities invest heavily in systems that track development and building permits. In Clark County, staff did not view the permit tracking system as a base upon which buildable lands data could be generated. There was significant interaction with the GIS system, but no interaction with the permit tracking system as a vehicle that reports zoning, parcel size, development type, and date. The county wanted to use this powerful tool and leverage the investment for the benefit of the comprehensive planning process.

The county also took the opportunity to use a project team that mixed public and private sector resources. Angelo Eaton and Associates was contracted to provide planning guidance for the project. They subcontracted with the county's Information Services Division to provide the technical expertise necessary to update the database, write reports, develop import scripts, etc. The Long Range Planning Division of the county managed the project. Additional county staff in building and administration added expertise for web development, data testing, and interaction with building officials. Grant funds were used for Long Range Planning, Information Services and consulting staff. Other county resources were provided as a supplement to the grant funds and were essential to the project's ultimate success and expansion in scope.

The project developed a buildable lands data collection process that accomplished the following project goals:

- Link the buildable lands data collection process with the Tidemark/Advantage permitting system.
- Link the data collection of individual communities to Tidemark\advantage, using techniques that do not require the communities to change existing permitting systems.
- Specify a common electronic protocol for data collection and transmission to Clark County.
- Create reports within each jurisdiction's systems that will extract the data ;
- Create a system easily maintained by jurisdictions.

In order to accomplish these goals, this project addressed the following four key program areas:

1. Develop data transfer protocols and a means of automatically importing data provided by other jurisdictions via Excel Spreadsheets into Tidemark/Advantage.
2. Develop buildable land reports in Tidemark/Advantage to use the available data.
3. Develop protocols for sharing Tidemark buildable lands reports via the county's website.
4. Develop a model system to provide local jurisdictions with lower cost access to Tidemark/Advantage using the City of Ridgefield as a case study.

DATA TRANSFER PROTOCOLS

As illustrated in Figure 1, data transfer protocols vary based on the jurisdiction. Building permit data for unincorporated Clark County (including the Urban Growth Areas) and the incorporated cities of Vancouver and Yacolt is maintained in Tidemark/Advantage. Data from each of the other jurisdictions will be transferred into Tidemark/Advantage "buildable lands" table via an Excel spreadsheet delivered to the county monthly via e-mail. Depending on the jurisdiction, the data is entered into the city generated Excel spreadsheet either through an automatic transfer from a permit tracking system or entered directly.

Figure 1: Data Transfer Protocols

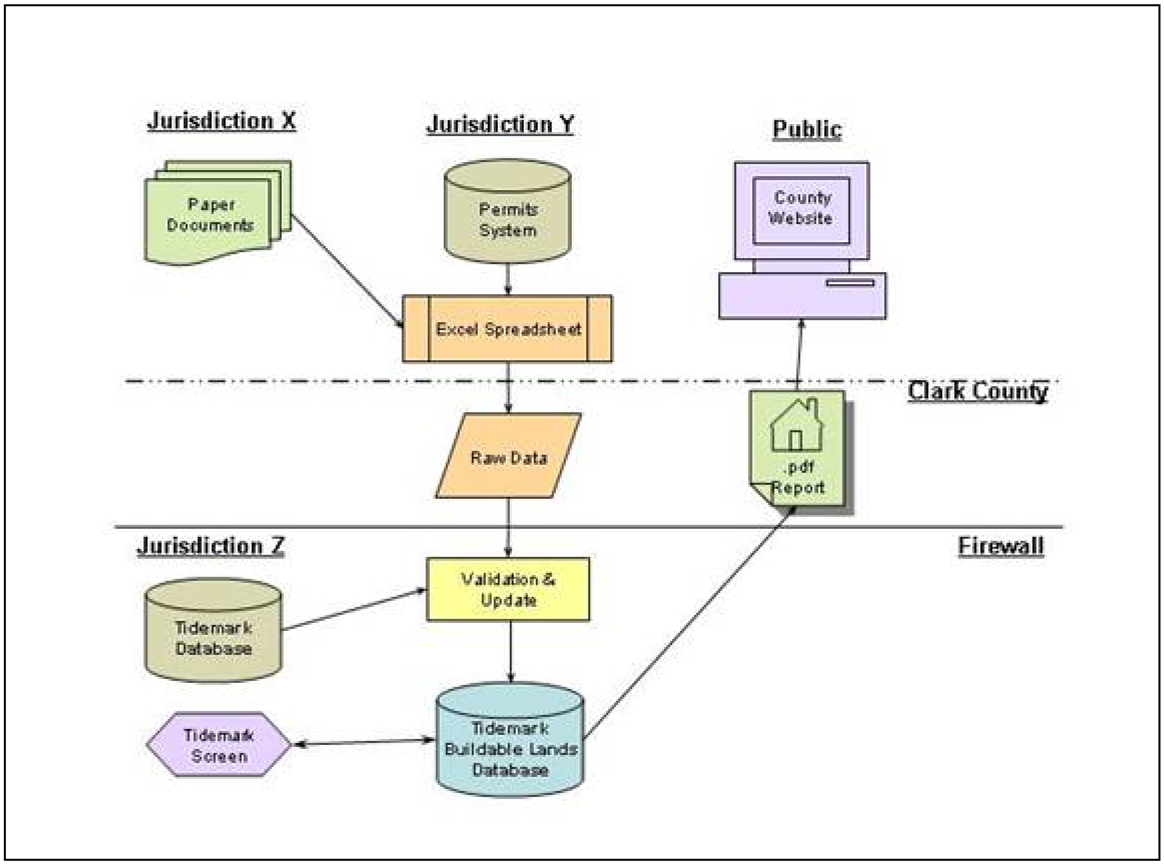


Table 1, below, summarizes the format for the standard Excel spreadsheet created by (or for) each jurisdiction. The following tables describe the source data by jurisdiction. In all cases, reported data includes:

- Only **approved** residential and non-residential building permits which create either:
  - **New** residential dwelling units, or
  - **New** commercial square footage,
  - **Parcel number,**
  - Address,
  - Date of permit issuance.

Once the data is imported into Tidemark, the county's Tidemark GIS link adds parcel size and percent critical area to each permit. Zoning is added also. These additions allow calculation of density. The GIS link also allows calculation showing the city and its attendant UGB. Tables 1a through 1h provide a detailed description of the data transfer protocols by jurisdiction.

Table 1
STANDARD FORMAT FOR DATA SPREADSHEET

Field Position	Field Name	Field Type & Size	Description	Data Source
1	Data Source	String, 4	Name of Jurisdiction or for unincorporated county specify UGB, if applicable	Autofill based on source of data or GIS based on primary parcel number
2	Permit No	String, 14	Permit or case number	Entered by jurisdiction
3	Permit Type	String, 3	<u>Residential Pick List</u> SFR Duplex Mobile Home MFR Mobile Home Park <u>Non-Residential Pick List</u> Commercial Retail Office Other Commercial Institutional Industrial	Entered by jurisdiction or auto-fill based on permit type
4	Date Issued	Date	Date of Permit Issuance	Entered by jurisdiction
5	Parcel No	String, 10	Assessor parcel number (Primary parcel only)	Entered by jurisdiction
6	No of Units	Number	Number of residential units created.	Entered by jurisdiction
7	Bldg SqFt	Number	Amount of non-residential square footage created.	Entered by jurisdiction
8	Owner	String, 30	Property owner name, if available	Entered by jurisdiction
9	Address	String, 30	Site or Construction Address, if available	Entered by jurisdiction
GIS	Lot SqFt	Number	Total square footage of all tax lots. *NOTE: multiple parcel data not available from all sources.*	Where development is on multiple parcels = Sum of area of all parcel
GIS	Zoning	String	City or county base zone	Based on Primary Parcel Number
GIS	Critical Area SqFt	Number	Sq. footage of critical areas	Based on Primary Parcel Number

Table 1a - Clark County			
System	Tidemark		
Able to provide data?	Yes, internal transfer		
Contact	Subu Swayam		
Data Fields:			
1	Data Source	Name of Jurisdiction – data auto-filled based on source = “CC”	
2	Permit No	csm_caseno (field format = “COM2003-00429”)	
3	Permit Type	SFR (attached and detached)	csm_case_type = SFR; where sfr_scope_of_work = all except DUPLEX and DUPLEX SA
		Duplex	csm_case_type = SFR; where sfr_scope_of_work = DUPLEX and DUPLEX SA
		Mobile Home (on individual lots)	csm_case_type = MOH; where moh_park_acreage = no; and moh_type_placement = permanent or mod;
		ADU	csm_case_type= RES; where csm_scope_of_work = all ADU options;
		Mobile home parks	csm_case_type = FSR; where csm_dev_type= mobile home park (new field to be added)
		MFR	csm_case_type = MFR; where csm_census_category = 104 or 105
		Commercial Retail	csm_case_type = COM; where csm_census_category = 322 or 327; and where csm_scope_of_work = new
		Office	csm_case_type = COM; where csm_census_category =324; and where csm_scope_of_work = NEW
		Institutional	csm_case_type = COM; where csm_census_category = 319, 323, 325, 326; and where csm_scope_of_work = NEW
		Industrial	csm_case_type = COM; where csm_census_category = 320; and where csm_scope_of_work = NEW
		Other Commercial	csm_case_type = COM; where csm_census_category = 213, 214, 318, 328; and where csm_scope_of_work = NEW
4	Date Issued	csm_issued_date	
5	Parcel No	prc_parcel_no (field format = “1871580000”)	
6	No of Units	value of csm_no_of_units; default 1 unit for ADU and MOH	
7	Bldg SqFt	case_valu.csv_item_amount	
8	Owner	case_people – role_type = “owner”	
9	Address	case_main – site_address	

Table 1b - Vancouver

System	Tidemark		
Able to provide data?	Yes, internal transfer		
Contact	Linda Devlin and Angie Potter		
		Data Fields:	
1	Data Source	Name of Jurisdiction – data auto-filled based on source = “VAN”	
2	Permit No	csm_caseno (field format = “SFR2003-00506”)	
3	Permit Type	SFR (attached and detached)	csm_case_type = SFR; where sfr_scope_of_work = all except DUPLEX and DUPLEX SA
		Duplex	csm_case_type = SFR; where sfr_scope_of_work = DUPLEX and DUPLEX SA
		Mobile Home (on individual lots)	csm_case_type = MOH; where moh_park_acreage = no; and moh_type_placement = permanent or mod;
		ADU	csm_case_type= RES; where csm_scope_of_work = all ADU options;
		Mobile home parks	csm_case_type = PSR; where csm_dev_type= mobile home park (new field to be added)
		MFR	csm_case_type = MFR; where csm_census_category = 104 or 105; where csm_no_of_units = not blank
		Commercial Retail	csm_case_type = COM; where csm_census_category = 322 or 327; and where csm_scope_of_work = new
		Office	csm_case_type = COM; where csm_census_category =324; and where csm_scope_of_work = NEW
		Institutional	csm_case_type = COM; where csm_census_category = 319, 323, 325, 326; and where csm_scope_of_work = NEW
		Industrial	csm_case_type = COM; where csm_census_category = 320; and where csm_scope_of_work = NEW
		Other Commercial	csm_case_type = COM; where csm_census_category = 213, 214, 318, 328; and where csm_scope_of_work = NEW
4	Date Issued	csm_issued_date	
5	Parcel No	prc_parcel_no (field format = “1871580000”)	
6	No of Units	value of csm_no_of_units; default 1 unit for ADU and MOH	
7	Bldg SqFt	case_valu.csv_item_amount	
8	Owner	case_people – role_type = “owner”	
9	Address	case_main – site_address	

Table 1c - Yacolt

System	Tidemark		
Able to provide data?	Yes, internal transfer		
Contact	Subu Swayam		
Data Fields:			
1	Data Source	Name of Jurisdiction – data auto-filled based on source = “YAC”	
2	Permit No	csm_caseno (field format = “COM2003-00429”)	
3	Permit Type	SFR (attached and detached)	csm_case_type = SFR; where sfr_scope_of_work = all except DUPLEX and DUPLEX SA
		Duplex	csm_case_type = SFR; where sfr_scope_of_work = DUPLEX and DUPLEX SA
		Mobile Home (on individual lots)	csm_case_type = MOH; where moh_park_acreage = no; and moh_type_placement = permanent or mod;
		ADU	csm_case_type= RES; where csm_scope_of_work = all ADU options;
		Mobile home parks	Not Available
		MFR	csm_case_type = MFR; where csm_census_category = 104 or 105
		Commercial Retail	csm_case_type = COM; where csm_census_category = 322 or 327; and where csm_scope_of_work = new
		Office	csm_case_type = COM; where csm_census_category =324; and where csm_scope_of_work = NEW
		Institutional	csm_case_type = COM; where csm_census_category = 319, 323, 325, 326; and where csm_scope_of_work = NEW
		Industrial	csm_case_type = COM; where csm_census_category = 320; and where csm_scope_of_work = NEW
		Other Commercial	csm_case_type = COM; where csm_census_category = 213, 214, 318, 328; and where csm_scope_of_work = NEW
4	Date Issued	csm_issued_date	
5	Parcel No	prc_parcel_no (field format = “1871580000”)	
6	No of Units	value of csm_no_of_units; default 1 unit for ADU and MOH	
7	Bldg SqFt	case_valu.csv_item_amount	
8	Owner	case_people – role_type = “owner”	
9	Address	case_main – site_address	

Table 1d – Battleground

System	Current permit tracking software does not provide the ability to prepare custom reports; however, since 2004, staff has also maintained a list of building permit activity in Excel to prepare a monthly report to the City Council. The City added the additional fields as requested. The City's Excel spreadsheet now includes the following fields: Project, **Location**, **Date Issued**, **Parcel Number**, **Permit Number**, **Commercial sf**., Residential sf., Garage sf, **Type**, Value, **Dwelling units**.
Able to provide data?	Yes, external transfer via emailed Excel spreadsheet
Contact	Rosemary Armour (building permit data) and Don Oehler (IT)

Data Fields:

1	Data Source	Name of Jurisdiction – data auto-filled based on source = "BG"	
2	Permit No	"Permit Number" field – "04-1016"	
3	Permit Type	SFR (attached and detached)	"Type" field = "New SFR"
		Duplex	"Type" field = "New Duplex"
		Mobile Home (on individual lots)	Not Available
		ADU	Not Available
		Mobile home parks	Not Available
		MFR	"Type" field = "New Multi-Family"
		Commercial Retail	Not Available – see default "Other Commercial"
		Office	Not Available – see default "Other Commercial"
		Institutional	Not Available – see default "Other Commercial"
		Industrial	Not Available – see default "Other Commercial"
		Other Commercial	Type field = "New Commercial"
4	Date Issued	"Date Issued" field – MM/DD/YYYY (e.g., "4/14/2005")	
5	Parcel No	"Parcel Number" field – "091057-003"	
6	No of Units	"Type" field = "New SFR" use default value of "1" "Type" field = "New duplex" use default value of "2" "Type" field = "New Multi-Family" use value in "Dwelling units" field	
7	Bldg SqFt	"Type" field = "New Commercial" use value in "Commercial sf." field	
8	Owner	Not available	
9	Address	"Location" field – "512 NW 22nd Avenue"	

Table 1e - Camas

System	Building permit tracking software
Able to provide data?	Yes, external transfer via emailed Excel spreadsheet, with the exception of commercial square footage (although that field could be added). Use only WorkType = “New”
Contact	Jeanette Kreighbaum

Data Fields:

1	Data Source	Name of Jurisdiction – data auto-filled based on source = “CAM”	
2	Permit No	“PermitNo” field – “02-18425”	
3	Permit Type	SFR (attached and detached)	“BuildingUse” field = “SINGLE-FAMILY” or “Building Use” field = “ROW HOUSE”
		Duplex	“BuildingUse” field = “TWO-FAMILY”
		Mobile Home (on individual lots)	“BuildingUse” field = “MOBILE HOME”
		ADU	Not available - see default “SFR”
		Mobile home parks	Not available - see default “SFR”
		MFR	“BuildingUse” field = “MULTI-FAMILY” “BuildingUse” field = “CONDOMINIUM/COOP”
		Commercial Retail	“FedNo” field = 322 or 327
		Office	“FedNo” field = 324
		Institutional	“FedNo” field = 319, 323, 325, 326
		Industrial	“FedNo” field = 320
		Other Commercial	“FedNo” field = 213, 214, 318, 328
4	Date Issued	“PermitDate” – MM/DD/YYYY (e.g., 7/29/2003)	
5	Parcel No	“TaxAccount” – “085200000”	
6	No of Units	Use value in “Units” field, “New SFR” use default value of “1”	
7	Bldg SqFt	Use value in “FMsqft” field	
8	Owner	Not available	
9	Address	Not available	

Table 1f – La Center

System	Building permit tracking software
Able to provide data?	Yes, external transfer via emailed Excel spreadsheet, the city has agreed to begin using fields as shown in Table 1 beginning in 2005.
Contact	Suzanne Levis and Alice Peters

Data Fields:

1	Data Source	Name of Jurisdiction – data auto-filled based on source = "LAC"	
2	Permit No	"Permit No" field – "RN04-395" RN- Residential New RR- Residential Remodel M- Mechanical P- Plumbing SP- Special Project CN- Commercial New CR- Commercial Remodel	
3	Permit Type	SFR (attached and detached)	"Permit No" field – "RN##-###"
		Duplex	Currently not available - see default "SFR"
		Mobile Home (on individual lots)	Currently not available - see default "SFR"
		ADU	Currently not available - see default "SFR"
		Mobile home parks	Currently not available - see default "SFR"
		MFR	Currently not available - see default "SFR"
		Commercial Retail	Currently not available – see default "Other Commercial"
		Office	Currently not available – see default "Other Commercial"
		Institutional	Currently not available – see default "Other Commercial"
		Industrial	Currently not available – see default "Other Commercial"
		Other Commercial	"Permit No." field = "CN##-###"
4	Date Issued	"Date Rec." field - MM/DD/YYYY (e.g., "8/17/2004")	
5	Parcel No	"TaxAccount" – "085200000"	
6	No of Units	Use default value of 1, City will be adding a units field	
7	Bldg SqFt	Use default value of 0, City will be adding a square footage field.	
8	Owner	"Applicant" field	
9	Address	"Address" field	

Table 1g - Ridgefield

System	Black Bear System
Able to provide data?	Yes, via emailed Excel spreadsheet
Contact	Joanne Hazen and Kevin Snyder

Data Fields:

1	Data Source	Name of Jurisdiction – data auto-filled based on source = “RDG”	
2	Permit No	“Permit Activity No.” field = “981-04” (field format = ###-YY)	
3	Permit Type	SFR (attached and detached)	“Resid. Permit Type” field = “SF”
		Duplex	“Resid. Permit Type” field = “MF”; where “MF Units created” = 2
		Mobile Home (on individual lots)	Currently not available - see default “SFR”
		ADU	Currently not available - see default “SFR”
		Mobile home parks	Currently not available - see default “SFR”
		MFR	“Resid. Permit Type” field = “MF”; where “MF Units created” = more than 2
		Commercial Retail	Currently not available – see default “Other Commercial”
		Office	Currently not available – see default “Other Commercial”
		Institutional	Currently not available – see default “Other Commercial”
		Industrial	Currently not available – see default “Other Commercial”
		Other Commercial	“Resid. Permit Type” field = “C”
4	Date Issued	“Issued Date” field = MM/DD/YYYY (e.g., 2/7/2005)	
5	Parcel No	Parcel Serial No. (field format = “123002-086”)	
6	No of Units	Sum of values from “SF Units created” and “MF Units created”; use default value of “1” for SF	
7	Bldg SqFt	Currently not available, use default value of 0	
8	Owner	“Property Owner” field	
9	Address	“Construction Address” field	

Table 1h - Washougal

System	Black Bear System
Able to provide data?	Yes, via emailed Excel spreadsheet
Contact	Mark Rauer and Rose Jewell

Data Fields:

1	Data Source	Name of Jurisdiction – data auto-filled based on source = “WAS”	
2	Permit No	“Permit Activity No.” field = “981-04” (field format = ###-YY)	
3	Permit Type	SFR (attached and detached)	“Resid. Permit Type” field = “SF”
		Duplex	“Resid. Permit Type” field = “MF”; where “MF Units created” = 2
		Mobile Home (on individual lots)	Currently not available - see default “SFR”
		ADU	Currently not available - see default “SFR”
		Mobile home parks	Currently not available - see default “SFR”
		MFR	“Resid. Permit Type” field = “MF”; where “MF Units created” = more than 2
		Commercial Retail	Currently not available – see default “Other Commercial”
		Office	Currently not available – see default “Other Commercial”
		Institutional	Currently not available – see default “Other Commercial”
		Industrial	Currently not available – see default “Other Commercial”
		Other Commercial	“Resid. Permit Type” field = “C”
4	Date Issued	“Issued Date” field = MM/DD/YYYY (e.g., 2/7/2005)	
5	Parcel No	Parcel Serial No. (field format = “123002-086”)	
6	No of Units	Sum of values from “SF Units created” and “MF Units created”; use default value of “1” for SF	
7	Bldg SqFt	Currently not available, use default value of 0	
8	Owner	“Property Owner” field	
9	Address	“Construction Address” field	

BUILDABLE LAND REPORTS

The following three standard reports have been developed in Tidemark using Crystal Reports:

- Residential and Non-Residential Detail Report
- Residential Summary Report
- Non-Residential Summary Report

RESIDENTIAL AND NON-RESIDENTIAL DETAIL REPORT

The *Detail Report* provides a complete record of all of the permit data in the database for either a selected month or a selected year. The data is grouped by "Jurisdiction" and then by "Category", within these groups, the data is sorted by "Permit Number". Subtotals of the number of dwelling units ("Units") and "Non-Residential Square Footage" are provided by "Jurisdiction", and a county-wide total is provided at the end. A sample report is included in the **Appendix A**.

RESIDENTIAL SUMMARY REPORT

The *Residential Summary Report* provides a summary of new single family and multi-family permits for a selected timeframe (month(s) or year(s)). Single family includes single family residential, accessory dwelling units (ADU), and mobile homes (on individual lots). Multi-family includes multi-family residential, duplexes, and new mobile home parks. Net Density is calculated based on the sum of the parcel square footage divided by the number of units. The "Percent Multi-Family" represents multi-family units as a percentage of total units. This report includes two charts, which are illustrated on the following page. "SFR" is the default residential type for those jurisdictions which do not provide more specific information in their reports. A sample report is included in the **Appendix B**.

NON-RESIDENTIAL SUMMARY REPORT

The *Non-Residential Summary Report* provides a summary of Non-Residential permits by development type for a selected timeframe (month(s) or year(s)). For each non-residential development type (Commercial Retail, Office, Institutional, Industrial, Other Commercial), both the number of permits issued and the area (in acres) are summarized. The total number and square footage of all development types is summarized by jurisdiction. Site acreage is the sum of the area of all subject parcels. Critical acres are also identified for all subject parcels. The Floor Area Ratio (FAR) is calculated by subtracting "Critical Areas" from "Site Acres", multiplying by 43,560, and then dividing "Total Square Footage" by this amount. This report includes two charts, which are illustrated on the following page. "Other Commercial/Not Specified" is the default commercial type for those jurisdictions which do not provide more specific information in their reports. A sample report is included in the **Appendix C**.

ADDITIONAL REPORTS

In addition to the reports described above, several supplemental reports are being developed and tested including: a Critical Areas Summary Report, a Summary of Data Reported by Month, and a Supplemental Detail Report.

- **Summary of Data Reported by Month.** The Summary of Data Reported by Month identifies which jurisdictions have reported data. Building permit data for the county (including all of the UGAs) and the cities of Vancouver and Yacolt is provided directly through Tidemark. This will be reported automatically and an affirmative response will be indicated in this report. The other cities will be responsible for reporting their building permit activity to the county on a monthly basis. There may be occasions where this information is delayed and, therefore, not reflected in the detail and summary reports. This report serves to inform users where there are gaps in the data, so that the numbers are not misinterpreted.

- **Critical Areas Summary Report.** The Critical Areas Summary report summarizes the number of residential and non-residential permits issued, the total parcel area of those parcels subject to a permit ("Parcel Acres"), and the area (in acres) of land designated as a Critical Area ("Critical Acres"). For the purposes of this analysis, Critical Areas include floodplains, steep slopes, wetlands, and hydric soils. The total amount of Critical Areas and the percentage of the total parcel acres that that represents are also shown.

- **Supplemental Detail Report.** The Supplemental Detail Report is supplemental to the Residential and Non-Residential Detail Report and similar in format, but only includes those records for which there was incomplete data (e.g., no parcel number was provided by the local jurisdiction).

ACCESSING REPORTS ON THE COUNTY'S WEBSITE

Clark County maintains a website for county data and reports. A copy of the reports described in the previous section will be generated as a PDF file on a monthly basis and posted to the county's website. This section describes the methodology for making reports accessible on the county's website. **Appendix D** includes a sample of the proposed web page.

- Add "Growth Management Plan Monitoring" as a new listing on the Data Library page

- Under "Growth Management Plan Monitoring" add:

  - About the Project - A brief description of the project as follows: The Growth Management Act requires Clark County to compile data that shows the progress of each community's comprehensive plan toward the goals of the Growth Management Act. Each community collects development data which is forwarded to the county and added to a central database. This website draws data from that database. It allows citizens, interest groups, elected officials and advisory boards the most comprehensive source of planning data. *The database and the website is made possible through the cooperation of each city and through a grant from the State of Washington Department of Community Trade and Economic Development.* Provide links to the reports:
    - Residential and Non-Residential Detail Report
    - Residential Summary Report
    - Non-Residential Summary Report

  - Hot topics - This section is intended to allow the planners to add discussion of certain kinds of data, analysis, etc.

  - Data collection methodology - A brief description of the methodology as follows:
    - Cities of Battle Ground, Camas, La Center, Ridgefield, and Washougal - Each month, staff in each jurisdiction forward an electronic spreadsheet to the county with updated development data. They forward permit types, parcel numbers, numbers of units, etc. The county staff adds the data into the county's Tidemark permitting system using an electronic system that eliminates the need to re-enter the data manually. County staff perform a visual check to confirm that the data has merged into the database correctly. The county runs another program that creates a report and a PDF file that is automatically placed on the web.
    - Clark County and the cities of Vancouver and Yacolt - The city of Vancouver and Clark County own the same permit processing software and share a database format and Clark County processes building permits for the City of Yacolt. Each month, staff runs a program that merges the data from Vancouver into the county database. The county runs another program that creates a report and a PDF file that is automatically placed on the web.

- Trouble shooting –
    - Data not available (NA) – the web text will indicate that this message means the data is not yet available. Even though cities send data monthly, there may be a lag of 45 to 60 days for each month's data because of the time it takes to compile and verify data. Check the city status table. It indicates the last month's submission by each community.
    - Pre 2000 data missing – the web text will indicate that data before the year 2000 may be missing of some cities because of inconsistencies in the way the data was compiled between communities. The county cannot address this situation. The city may be able to help.
    - Reports do not show the desired data – the web text will indicate two options are available. Call the Long Range Planning Division of DCD (#####) and ask if a different report can be generated. We may be able to show additional data. Please do not request reformatting. If the county cannot generate the report, call the city and see if they have the ability to generate the data. City staff and city officials should call Long Range Planning if additional reports are needed for use by the jurisdiction.
    - Reports – the web text will indicate all reports on the web are in PDF format. If customers have difficulty printing, they should check for a compatible version of Adobe Acrobat.
    - Data seems wrong – the web text will direct questions to Long Range Planning if customers believe the data on the web is not correct. Planning will check to make sure the data was not corrupted during the conversion process.

PROVIDING A TIDEMARK/ADVANTAGE CONNECTION

In the late 1990's the city of Vancouver and Clark County made the deliberate decision to create joint permitting databases. Looking forward to annexations, they concluded that a consistent approach to development and building activity would be easier for customers to navigate and easier for staff to administer. They transitioned to a product called Tidemark, which is now called Advantage by Accela. Internally, the county refers to the permitting system as Tidemark.

In this project, the county discussed permitting systems with the other cities. Some systems are old and some have insufficient features to deal with current workloads. Some communities have a permitting system, but cannot print reports and have resorted to a dual spreadsheet and permit system process. Several communities are thinking about replacing their systems. The county discussed the potential of using Tidemark in the other cites. Specific conversations were held in La Center and Washougal and with the Hazel Dell Sewer District. Staff generally discussed the concept in Ridgefield and Battle Ground. The county offered:

- System backups and database administration by professionals trained in Tidemark;
- Upgrades;
- A fully developed system including cases, reports, and links to other databases;
- Data that is comparable to Clark County and the city of Vancouver;
- A system that builders and developers know and understand; and
- Access to the Clark County computer network that will increase system reliability and provide back up support for City staff (optional).

Doing so could create benefits in lower costs, more consistent data, and a larger network of support through knowledgeable staff in several communities. Communities could share report formats, triggers and other tools, and save money. It would also make the plan monitoring data easier to compile because the data would import more easily.

Prior to 2004, the costs of having other cites join the city and county system was prohibitive because of networking and file transfer costs. For security reasons other communities would have had to join onto the county network and turn over their security to the county. Not only was this expensive because of providing fiber optic to communities and adding county network costs, it was politically unpalatable. Quite reasonably, communities did not want to give up control of their internal computer systems to the county.

In 2004, the county offered a terminal server (i.e. Citrix) approach (shown in Figure 2) that was more feasible. The terminal server acts as a "mirror" of the database. A community can use an internet connection to enter data into the terminal server mirror. The terminal server moves the data to the database over a secure connection. Discussions proposed that each community have a separate database for auditing purposes.

The new approach was more interesting to the communities and special districts because of lower costs and because the county could create separate databases for greater management control and eliminate the need for the community to give up control of their system. It is internet based, which allows access at any location. Fiber optic cable is not required.

Sample costs to set up one community, for example La Center are:

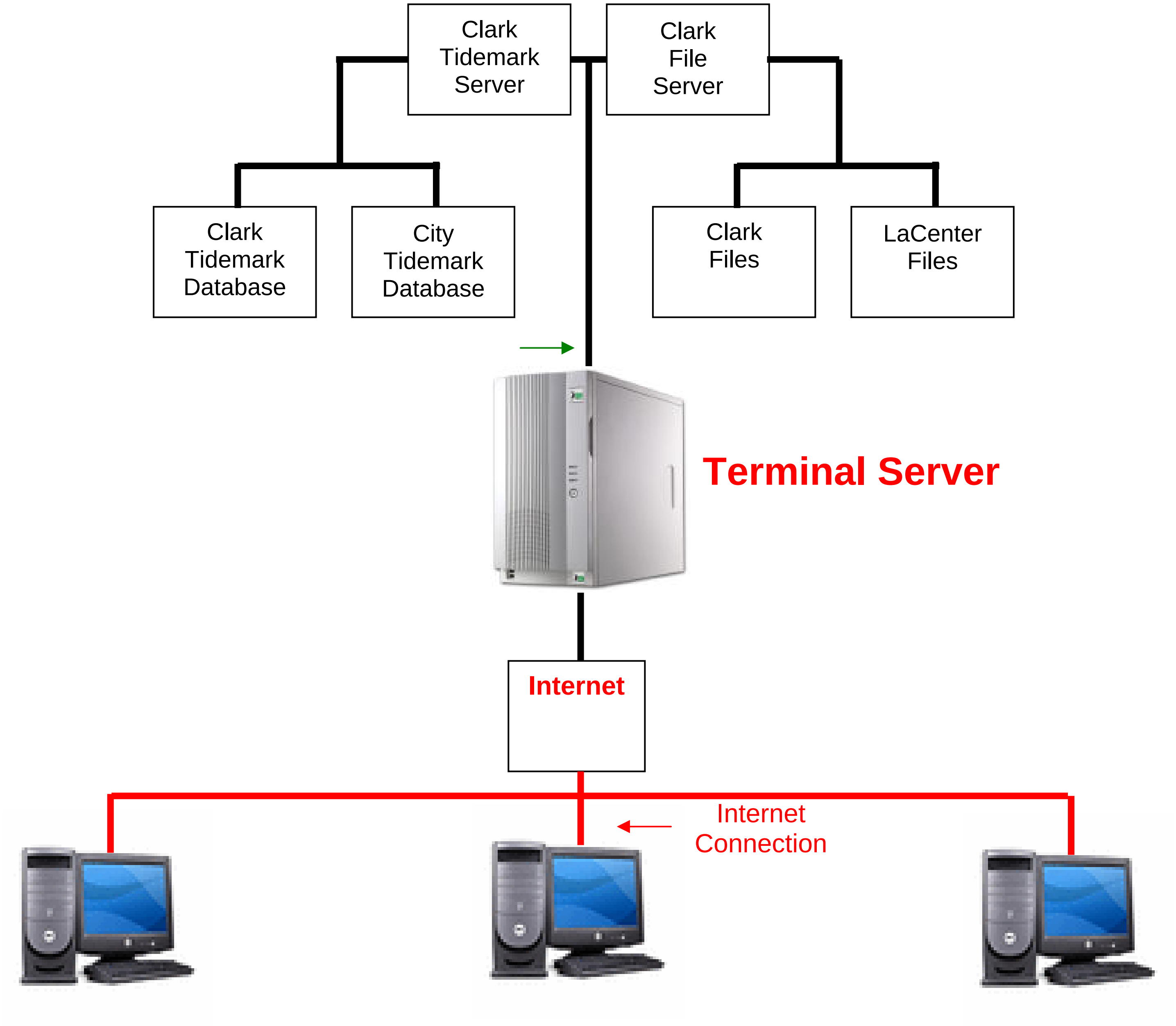
- One time costs: $50,000, including set up of databases, tailoring reports to the community, training, fee table development, licenses and servers.

- Continuing annual costs: $30,000 including licenses, support from the county, report updates, etc.

General discussions continue. Staff observes that communities that feel they cannot afford the continuing costs are likely to stay with excel spreadsheets or use a product called Springbrook that is accounting software with a building permit module added. Communities that feel they can afford a permitting system want to put their own "stamp" on the decision and are less interested in collaboration.

Figure 2

Citrix Version



LESSONS LEARNED AND NEXT STEPS

COORDINATION WITH OTHER DEPARTMENTS

- While monitoring the intensity and density of development and comparing growth and development assumptions, targets, and objectives is typically a task managed by planning departments, the necessary data (primarily building permits) is tracked and managed by building officials. Building officials are responsible for a wide variety of permit types (e.g., remodeling, demolitions, plumbing, electrical, etc.) and a wide range of data related to permit review, inspection and processing time. Isolating the specific types of building permits and data fields of use to monitoring growth can be a significant challenge. Getting the cooperation of building officials to forward the data can also be a challenge.

- In addition to planning and building, many different departments and agencies track or are interested in tracking building permit activity and require jurisdictions to provide some or all of that data. Competing demands for different subsets of the building permit data can lead to duplicative reporting and additional effort and expense for communities.

- As part of this project the Project Team met with County Assessor's staff to discuss the concept of using the Assessor's parcel data (either directly from CATs or via the GIS). The Assessor's office enters every county and city building permit that affects property value into their program. Key points for future consideration include:
  - Currently not all of the fields desired for the buildable lands tracking are entered. The Assessor's office will be updating their software and there may be an opportunity to have the desired fields included in the future.
  - There can be a significant lag in data entry, which may make the Assessor's data unsuitable for monthly reporting; however, as a source of historic annual data it has several advantages:
    - Assessor's data is in a consistent format for the county and all cities,
    - Assessor's data is reliable in terms of units actually constructed (not just permitted),
    - Property description field provides high level of detail about type of development (e.g., "Single family unit sharing a common wall, individual lot ownership"), and
    - Assessor's permit data is already associated with tax parcels (no additional address matching or data entry required)

- Coordination with Addressing – One of the key issues with the data that has been provided by the cities (with the exception of Camas) is a lack of tax parcel information. Because the information needed for the buildable lands analysis depends on this field, this is a significant shortcoming. County staff responsible for GIS and addressing explained that they are hoping to expedite the process of providing new parcel numbers to the cities. They will be working with Clark Public Utility to obtain information on new subdivisions early in the process and providing tentative parcel numbers to the cities. Their hope is that situs addresses and parcel numbers could be coordinated via a web interface. In discussing our joint data needs, an opportunity to establish a joint web interface/database was discussed. The concept includes the following key ideas:

- When issuing a building permit for a new dwelling or commercial building, jurisdictional staff would access a database on the web.
- The database would provide them with the subdivision name, lot number and tentative parcel number (if a newly recorded subdivision) or actual parcel number (if available for existing parcels).
- Jurisdictional staff would enter the situs address (or accept a suggested tentative address if available) into the database.
- Jurisdictional staff would also be required to enter the permit number, the number of units and the amount of commercial square footage.
- The advantage to the Buildable Lands Tracking project in teaming with Addressing include: reducing redundant data entry and elevating the status of the data entry (jurisdictional staff may be more likely to do this than to remember to email monthly permit data to the county).

CONTINUING COMMITMENT FOR DATA

The long term success of the buildable lands system relies upon cities forwarding data and the county importing it into the Tidemark system. In an attempt to institutionalize data transfer, the county will construct a letter of agreement for execution by each city, with the exception of Vancouver. The Vancouver data is accessible to the county because of the joint permitting system. The county will forward these agreements to cities in July, after the web site is operational.
The agreement will specify the following commitments by each city:

- Contact person
- Frequency of data transfer
- Format and data to be transferred
- Agreement to communicate any changes in format to the county.
- Agreement that failure to provide data may result in a per hour data collection charge by the county if the county has to manually collect data.

The agreement will specify the following commitments by the county:

- Contact person
- Frequency of data import into Tidemark
- Frequency of web site update
- Content of web site reports
- Process for requesting additional special reports

MARKETING PLAN

Clark County recognizes that cities will have greater impetus to provide data if the web site is used by the cities and other interest groups in the community. In July, the county plans the following marketing effort to create demand for the plan monitoring data

- **Marketing efforts:** The county will use an integrated marketing approach to create a demand for information among select target audiences and encourage local jurisdiction participation. The core processes for implementing marketing efforts include media relations, direct communications and electronic communications.

- **Media Relations:** The primary focus of the media strategy will be working with local print and broadcast news media including: press releases to local newspapers, advertisements and articles in trade and neighborhood association publications, and a recurring spot on Clark Vancouver Television (CVTV Channel 23).

- **Direct Communications:** Clark County will communicate project benefits and the availability of updated data through direct mail of informational flyers, reports with corresponding articles on emerging development trends, as well as telephone message alerts.

- **Electronic Communications:** Email message alerts will be used as notification of updated information and the availability of comprehensive reports. The Growth Plan Monitoring Web page will host up-to-date articles on emerging development trends as seen in the collected data. Emerging trend articles and site link will also be marketed through articles in eNews, the Clark County Community Development electronic publication.

APPENDIX A

SAMPLE RESIDENTIAL AND NON-RESIDENTIAL DETAIL REPORT

Permits Issued

Reporing Period: 1/1/2004 - 12/31/2005

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
Berryville						
Non-Residential						
1	05-228	04/29/2005	0910570420	Commercial Retail	0	2,561
Residential						
1	03-315	04/28/2005	2287540260	Single Family	1	0
2	04-1016	04/14/2005	0910570030	Multi-Family	10	0
3	04-1017	04/07/2005	0910570030	Multi-Family	12	0
4	04-999	04/19/2005	0910570030	Multi-Family	8	0
5	05-091	04/19/2005	2285731640	Single Family	1	0
6	05-185	04/01/2005	0910573080	Single Family	1	0
7	05-193	04/01/2005	0910572820	Single Family	1	0
8	05-220	04/15/2005	0910572740	Single Family	1	0
9	05-224	04/29/2005	0910572620	Single Family	1	0
10	05-225	04/29/2005	0910572860	Single Family	1	0
11	05-245	04/19/2005	0910573060	Single Family	1	0
12	05-275	04/18/2005	1921410660	Single Family	1	0
13	05-280	04/19/2005	1921410080	Single Family	1	0
14	05-281	04/25/2005	1921410920	Single Family	1	0
15	05-282	04/25/2005	1921410640	Single Family	1	0
16	05-283	04/19/2005	1921410860	Single Family	1	0
17	05-299	04/28/2005	1921411140	Single Family	1	0
18	05-301	04/28/2005	1921411120	Single Family	1	0
19	05-316	04/29/2005	1921410620	Single Family	1	0
20	05-331	04/29/2005	1921410380	Single Family	1	0
21	05-332	04/29/2005	1921410400	Single Family	1	0
22	05-348	04/29/2005	1921410520	Single Family	1	0
23	05-349	04/29/2005	1921410020	Single Family	1	0
				Sub Total	50	2,561
Berryville UGA						
Non-Residential						
1	SubuMadeUp-001	01/01/2005	1192021700	Other Commertial	0	3,456
2	SubuMadeUp-002	01/01/2005	1192021240	Other Commertial	0	2,345
Residential						
1	SubuMadeUp-003	01/01/2005	1192022200	Mobile Home Park	40	0
				Sub Total	40	5,801

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
Milltown						
Non-Residential						
1	05-21762	01/05/2005	1275022640	Other Commertial	0	908
2	05-21776	01/11/2005	0813500000	Other Commertial	0	550
3	05-21813	01/26/2005	0845500000	Other Commertial	0	864
4	05-21814	01/27/2005	0872100000	Office	0	1,164
5	05-21915	02/28/2005	0794700000	Commercial Retail	0	1,240
6	05-21969	03/14/2005	0909930000	Office	0	2,031
7	05-22069	04/08/2005	0878600000	Industrial	0	672
Residential						
1	04-21757	01/05/2005	1250770220	Single Family	1	0
2	05-21760	01/05/2005	1253940220	Single Family	1	0
3	05-21768	01/07/2005	1101869600	Single Family	1	0
4	05-21772	01/10/2005	0851440120	Single Family	1	0
5	05-21774	01/10/2005	1101865620	Single Family	1	0
6	05-21778	01/11/2005	1238340200	Single Family	1	0
7	05-21791	01/20/2005	1248172040	Single Family	1	0
8	05-21793	01/20/2005	1248172340	Single Family	1	0
9	05-21795	01/20/2005	1248172180	Single Family	1	0
10	05-21797	01/20/2005	1759320120	Single Family	1	0
11	05-21801	01/21/2005	1248171600	Single Family	1	0
12	05-21808	01/26/2005	1248171760	Single Family	1	0
13	05-21811	01/26/2005	1248171620	Single Family	1	0
14	05-21817	01/28/2005	1248171660	Duplex	2	0
15	05-21822	01/28/2005	1248172440	Duplex	2	0
16	05-21824	01/28/2005	1248172460	Duplex	2	0
17	05-21832	02/03/2005	1248171700	Duplex	2	0
18	05-21835	02/03/2005	1253940420	Duplex	2	0
19	05-21838	02/08/2005	1248172020	Duplex	2	0
20	05-21840	02/08/2005	1248172240	Duplex	2	0
21	05-21842	02/08/2005	1248171840	Duplex	2	0
22	05-21845	02/09/2005	1248172280	Duplex	2	0
23	05-21853	02/14/2005	1248172380	Duplex	2	0
24	05-21858	02/14/2005	1248172480	Duplex	2	0
25	05-21865	02/15/2005	1273690260	Duplex	2	0
26	05-21868	02/16/2005	1253940300	Duplex	2	0
27	05-21870	02/16/2005	1253940160	Duplex	2	0
28	05-21872	02/16/2005	1248170760	Duplex	2	0
29	05-21879	02/17/2005	1101865280	Duplex	2	0
30	05-21881	02/17/2005	1249860140	Duplex	2	0
31	05-21887	02/22/2005	1253940440	Duplex	2	0
32	05-21892	02/23/2005	1248170800	Duplex	2	0
33	05-21894	02/23/2005	0837790480	Duplex	2	0
34	05-21911	02/25/2005	1250082200	Duplex	2	0
35	05-21918	03/01/2005	1248170780	Single Family	1	0
36	05-21931	03/04/2005	1249690200	Single Family	1	0
37	05-21934	03/07/2005	0819591740	Multi-Family	6	0
38	05-21936	03/07/2005	0819591760	Multi-Family	12	0
39	05-21938	03/07/2005	0819591700	Multi-Family	8	0
40	05-21940	03/07/2005	0819591720	Multi-Family	10	0
41	05-21942	03/07/2005	1248171240	Single Family	1	0
42	05-21944	03/07/2005	1248171920	Single Family	1	0
43	05-21946	03/08/2005	1248172320	Mobile Home Park	64	0
44	05-21950	03/09/2005	1248171640	Mobile Home Park	12	0
45	05-21973	03/15/2005	1273690760	Mobile Home	1	0
46	05-21985	03/21/2005	1248171900	Mobile Home	1	0
47	05-21988	03/22/2005	1248171880	Mobile Home	1	0
48	05-21999	03/24/2005	1273690160	Mobile Home	1	0
49	05-22001	03/24/2005	0898620300	Mobile Home	1	0
50	05-22006	03/28/2005	1238310240	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
51	05-22008	03/28/2005	1238310320	Single Family	1	0
52	05-22010	03/28/2005	1238310400	Single Family	1	0
53	05-22012	03/28/2005	1238310700	Single Family	1	0
54	05-22014	03/28/2005	1238310720	Single Family	1	0
55	05-22016	03/28/2005	1238310740	Single Family	1	0
56	05-22020	03/28/2005	1238310820	Single Family	1	0
57	05-22022	03/28/2005	1238310880	Single Family	1	0
58	05-22025	03/28/2005	1238310900	Single Family	1	0
59	05-22027	03/28/2005	1238310920	Mobile Home	1	0
60	05-22032	03/28/2005	0807300000	Mobile Home	1	0
61	05-22034	03/28/2005	0807250000	Mobile Home	1	0
62	05-22040	03/29/2005	0922327260	Single Family	1	0
63	05-22046	03/31/2005	1248171720	Single Family	1	0
64	05-22050	04/04/2005	1253940400	Single Family	1	0
65	05-22057	04/05/2005	0873680380	Single Family	1	0
66	05-22061	04/06/2005	0910458960	Additions	0	0
67	05-22067	04/08/2005	1101865340	Single Family	1	0
				Sub Total	193	7,429
Milltown UGA						
Non-Residential						
1	SubuMadeUp-004	01/01/2005	1781740000	Industrial	0	1,234
2	SubuMadeUp-005	01/01/2005	1781180000	Commercial Retail	0	4,321
Residential						
1	SFR2004-00858	06/16/2004	1256060560	Single Family	1	0
				Sub Total	1	5,555

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
La Casino						
Non-Residential						
1	RN04-366	03/17/2004	2589920340	Commercial Retail	0	9,876
2	RN04-367	04/15/2004	0626910050	Commercial Retail	0	8,765
3	RN04-370	04/15/2004	0644200100	Commercial Retail	0	7,654
4	RN04-372	04/29/2004	2589910460	Commercial Retail	0	6,543
5	RN04-373	04/29/2004	2589910440	Commercial Retail	0	5,432
6	RN04-374	04/29/2004	2589910680	Office	0	4,321
7	RN04-377	06/17/2004	2589930180	Office	0	1,234
8	RN04-378	06/17/2004	0632700000	Office	0	2,345
9	RN04-379	06/17/2004	0634723900	Office	0	3,456
10	RN04-384	07/02/2004	2589930200	Office	0	4,567
11	RN04-386	07/02/2004	2589930260	Industrial	0	5,678
12	RN04-395	10/20/2004	2589920220	Industrial	0	6,789
13	RN04-401	10/21/2004	2589930300	Industrial	0	7,890
14	RN04-408	11/23/2004	0634729420	Industrial	0	1,357
15	RN04-409	12/02/2004	0634728440	Industrial	0	2,468
16	RN04-410	12/03/2004	0634728420	Other Commertial	0	3,579
17	RN04-411	12/06/2004	0634728760	Other Commertial	0	4,680
Residential						
1	RN04-317	03/08/2004	0626420300	Single Family	1	0
2	RN04-318	03/08/2004	0634721820	Single Family	1	0
3	RN04-325	03/08/2004	2589910480	Single Family	1	0
4	RN04-326	03/08/2004	0634724720	Single Family	1	0
5	RN04-332	03/08/2004	0634600000	Single Family	1	0
6	RN04-333	03/08/2004	2589910660	Single Family	1	0
7	RN04-334	03/08/2004	2589910560	Single Family	1	0
8	RN04-335	03/08/2004	2589910520	Single Family	1	0
9	RN04-336	03/08/2004	2589910620	Single Family	1	0
10	RN04-339	03/08/2004	2589910180	Single Family	1	0
11	RN04-340	03/08/2004	2589910640	Single Family	1	0
12	RN04-341	03/08/2004	2589911240	Single Family	1	0
13	RN04-342	03/08/2004	2589911280	Single Family	1	0
14	RN04-343	03/08/2004	2589911300	Single Family	1	0
15	RN04-344	03/08/2004	2589920360	Single Family	1	0
16	RN04-345	03/08/2004	2589910580	Single Family	1	0
17	RN04-346	03/08/2004	2589920140	Single Family	1	0
18	RN04-347	03/08/2004	2589920240	Single Family	1	0
19	RN04-348	03/08/2004	2589910100	Duplex	2	0
20	RN04-350	03/08/2004	0634724700	Duplex	2	0
21	RN04-351	03/08/2004	2589920100	Mobile Home	1	0
22	RN04-356	03/08/2004	0634726640	Multi-Family	20	0
23	RN04-357	03/08/2004	2589911380	Multi-Family	10	0
24	RN04-358	03/08/2004	2589910400	Multi-Family	8	0
25	RN04-360	03/16/2004	2589911360	Multi-Family	8	0
26	RN04-362	03/19/2004	0634721800	Multi-Family	4	0
27	RN04-363	03/25/2004	0634724920	Multi-Family	4	0
				Sub Total	77	86,634
La Casino UGA						
Non-Residential						
1	SubuMadeUp-006	01/01/2005	2587630000	Industrial	0	5,432
2	SubuMadeUp-007	01/01/2005	2587600000	Institutional	0	6,543
3	SubuMadeUp-008	01/01/2005	2587640000	Commercial Retail	0	7,654
Residential						
1	SubuMadeUp-009	01/01/2005	2587040000	Duplex	2	0
2	SubuMadeUp-010	01/01/2005	2586360000	Multi-Family	8	0
				Sub Total	10	19,629

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
Heron City						
Non-Residential						
1	273-02	09/11/2004	2140370000	Other Commertial	0	9,648
2	346-02	03/18/2005	2139790000	Other Commertial	0	5,367
3	359-02	05/02/2005	2139790000	Other Commertial	0	12,991
4	435-02	10/25/2004	0683140000	Other Commertial	0	4,849
5	475-03	05/08/2005	2142180000	Other Commertial	0	1,350
6	753-04	04/22/2005	0683180420	Commercial Retail	0	1,890
7	754-04	05/20/2005	0673690540	Commercial Retail	0	3,456
8	755-04	05/10/2005	0683180260	Office	0	1,950
9	756-04	04/30/2005	2135130320	Office	0	1,850
10	757-04	04/30/2005	2135130660	Office	0	3,860
11	758-04	05/06/2005	2135130800	Institutional	0	4,896
12	759-04	05/06/2005	2135130280	Institutional	0	65,432
13	763-04	05/07/2005	2135130540	Industrial	0	9,876
14	764-04	05/07/2005	2135130880	Industrial	0	19,000
15	766-04	05/14/2005	2135130960	Industrial	0	27,500
Residential						
1	365-02	06/03/2005	0709700000	Multi-Family	2	0
2	733-04	04/02/2005	2135130600	Single Family	1	0
3	736-04	04/02/2005	2135131220	Single Family	1	0
4	737-04	04/07/2005	2135131260	Single Family	1	0
5	740-04	04/14/2005	2135131120	Single Family	1	0
6	741-04	04/07/2005	2135130780	Single Family	1	0
7	744-04	04/14/2005	2135131020	Single Family	1	0
8	745-04	04/14/2005	2135130400	Single Family	1	0
9	746-04	04/14/2005	2135130060	Single Family	1	0
10	747-04	06/14/2004	0683180340	Single Family	1	0
11	748-04	06/14/2004	0683180880	Mobile Home	1	0
12	749-04	04/29/2005	2135130160	Mobile Home	1	0
13	750-04	04/29/2005	2135131160	Duplex	2	0
14	751-04	07/14/2004	0704150000	Single Family	1	0
15	752-04	07/20/2004	0691900000	Single Family	1	0
16	767-04	05/14/2005	2135130580	Multi-Family	4	0
17	770-04	06/08/2004	0683180680	Multi-Family	6	0
18	772-04	05/20/2005	2135133700	Mobile Home Park	20	0
19	773-04	05/20/2005	2135133100	Multi-Family	8	0
20	774-04	05/19/2005	2135130980	Multi-Family	8	0
21	775-04	05/19/2005	2135131040	Multi-Family	16	0
				Sub Total	79	173,915
Heron City UGA						
Non-Residential						
1	SubuMadeUp-011	01/01/2005	2158430000	Office	0	8,765
2	SubuMadeUp-012	01/01/2005	2158460000	Commercial Retail	0	9,876
Residential						
1	SubuMadeUp-013	01/01/2005	2158340000	Multi-Family	6	0
				Sub Total	6	18,641

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
Gromania						
Non-Residential						
1	COM2004-00251	06/23/2004	1098950320	Office	0	1,659
2	COM2004-00252	06/23/2004	1098950340	Office	0	2,468
3	COM2004-00253	06/23/2004	1098950300	Commercial Retail	0	1,357
4	COM2004-00254	06/23/2004	1098950240	Institutional	0	9,753
5	COM2004-00256	06/30/2004	1632150920	Office	0	6,420
Residential						
1	MFR2003-00070	04/14/2004	1583540000	Multi-Family	9	0
2	MFR2003-00071	04/20/2004	1583540000	Multi-Family	9	0
3	MOH2004-00007	05/27/2004	1097743560	Mobile Home	1	0
4	SFR2003-00506	04/16/2004	1647860520	Single Family	1	0
5	SFR2003-00531	06/08/2004	1616570440	Single Family	1	0
6	SFR2003-00533	06/08/2004	1616570460	Single Family	1	0
7	SFR2003-00535	06/08/2004	1616570480	Single Family	1	0
8	SFR2003-00536	06/08/2004	1616570500	Single Family	1	0
9	SFR2003-00537	06/08/2004	1616570520	Single Family	1	0
10	SFR2003-00538	06/08/2004	1616570540	Single Family	1	0
11	SFR2003-00539	06/08/2004	1616570560	Single Family	1	0
12	SFR2003-00540	06/08/2004	1616570580	Single Family	1	0
13	SFR2003-00541	06/08/2004	1616570600	Single Family	1	0
14	SFR2003-00548	06/08/2004	1616570740	Single Family	1	0
15	SFR2003-00549	06/08/2004	1616570760	Single Family	1	0
16	SFR2003-00550	06/08/2004	1616570780	Single Family	1	0
17	SFR2003-00551	06/08/2004	1616570800	Single Family	1	0
18	SFR2003-00556	05/04/2004	1647010120	Single Family	1	0
19	SFR2004-00002	04/02/2004	1647010340	Single Family	1	0
20	SFR2004-00005	06/02/2004	0920097860	Single Family	1	0
21	SFR2004-00024	04/27/2004	1647010400	Single Family	1	0
22	SFR2004-00025	04/01/2004	1221560100	Single Family	1	0
23	SFR2004-00041	04/28/2004	1647010840	Single Family	1	0
24	SFR2004-00044	04/21/2004	1647010960	Single Family	1	0
25	SFR2004-00046	06/08/2004	1669830000	Single Family	1	0
26	SFR2004-00047	05/21/2004	0379142120	Single Family	1	0
27	SFR2004-00048	04/12/2004	1647010600	Single Family	1	0
28	SFR2004-00058	04/09/2004	1647860680	Single Family	1	0
29	SFR2004-00060	04/05/2004	0379111470	Single Family	1	0
30	SFR2004-00061	05/10/2004	1647860920	Single Family	1	0
31	SFR2004-00063	05/14/2004	1647860440	Single Family	1	0
32	SFR2004-00064	06/28/2004	1647860600	Single Family	1	0
33	SFR2004-00065	04/05/2004	1593490120	Single Family	1	0
34	SFR2004-00066	04/13/2004	1593490140	Single Family	1	0
35	SFR2004-00092	04/08/2004	1084900180	Single Family	1	0
36	SFR2004-00093	05/20/2004	1647010640	Single Family	1	0
37	SFR2004-00094	04/15/2004	0352400000	Duplex	2	0
38	SFR2004-00095	04/19/2004	1099100240	Single Family	1	0
39	SFR2004-00096	05/28/2004	1647010060	Single Family	1	0
40	SFR2004-00097	04/16/2004	1647860800	Single Family	1	0
41	SFR2004-00098	04/01/2004	1632150440	Single Family	1	0
42	SFR2004-00100	04/01/2004	1632150460	Single Family	1	0
43	SFR2004-00104	05/07/2004	1647010080	Single Family	1	0
44	SFR2004-00105	04/30/2004	1674290700	Single Family	1	0
45	SFR2004-00106	04/30/2004	1674290060	Single Family	1	0
46	SFR2004-00107	05/14/2004	1632150700	Single Family	1	0
47	SFR2004-00116	06/30/2004	1632150180	Single Family	1	0
48	SFR2004-00117	05/20/2004	1647010860	Single Family	1	0
49	SFR2004-00118	04/26/2004	1221560120	Single Family	1	0
50	SFR2004-00119	06/14/2004	1124273320	Single Family	1	0
51	SFR2004-00120	04/02/2004	1100871880	Single Family	1	0
52	SFR2004-00121	06/01/2004	1100871780	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
53	SFR2004-00122	04/02/2004	1100871900	Single Family	1	0
54	SFR2004-00123	06/01/2004	1100871800	Single Family	1	0
55	SFR2004-00124	04/02/2004	1100871920	Single Family	1	0
56	SFR2004-00125	04/02/2004	1100871860	Single Family	1	0
57	SFR2004-00126	06/01/2004	1100871820	Single Family	1	0
58	SFR2004-00127	06/03/2004	1100871840	Single Family	1	0
59	SFR2004-00128	04/02/2004	1266840500	Single Family	1	0
60	SFR2004-00129	04/02/2004	1266840760	Single Family	1	0
61	SFR2004-00130	04/02/2004	1266841120	Single Family	1	0
62	SFR2004-00131	04/30/2004	1775010060	Single Family	1	0
63	SFR2004-00132	05/07/2004	1138930040	Single Family	1	0
64	SFR2004-00133	06/09/2004	1647010100	Single Family	1	0
65	SFR2004-00134	05/03/2004	0064500000	Duplex	2	0
66	SFR2004-00135	05/25/2004	1097783340	Single Family	1	0
67	SFR2004-00137	05/26/2004	1097783280	Single Family	1	0
68	SFR2004-00138	04/12/2004	1266840120	Single Family	1	0
69	SFR2004-00139	04/02/2004	1266840080	Single Family	1	0
70	SFR2004-00140	05/28/2004	1670070000	Single Family	1	0
71	SFR2004-00141	05/24/2004	1125200000	Single Family	1	0
72	SFR2004-00143	06/01/2004	0919050000	Single Family	1	0
73	SFR2004-00144	06/07/2004	1132100000	Single Family	1	0
74	SFR2004-00145	06/10/2004	1098950260	Single Family	1	0
75	SFR2004-00146	06/10/2004	1098950280	Single Family	1	0
76	SFR2004-00147	06/08/2004	1098950120	Single Family	1	0
77	SFR2004-00148	06/08/2004	1098950100	Single Family	1	0
78	SFR2004-00149	05/04/2004	1266841100	Single Family	1	0
79	SFR2004-00150	05/14/2004	1632150680	Single Family	1	0
80	SFR2004-00151	05/07/2004	1110226160	Single Family	1	0
81	SFR2004-00152	05/04/2004	1266841160	Single Family	1	0
82	SFR2004-00153	05/04/2004	1266841060	Single Family	1	0
83	SFR2004-00154	06/29/2004	1774964300	Single Family	1	0
84	SFR2004-00158	06/11/2004	1774964140	Single Family	1	0
85	SFR2004-00160	06/08/2004	1774964260	Single Family	1	0
86	SFR2004-00161	05/24/2004	1674290640	Single Family	1	0
87	SFR2004-00162	05/24/2004	1674290160	Single Family	1	0
88	SFR2004-00163	06/17/2004	1674290740	Single Family	1	0
89	SFR2004-00164	06/04/2004	1674290620	Single Family	1	0
90	SFR2004-00165	06/04/2004	1674290080	Single Family	1	0
91	SFR2004-00166	06/04/2004	1674290600	Single Family	1	0
92	SFR2004-00167	06/08/2004	1110225980	Single Family	1	0
93	SFR2004-00168	05/21/2004	1632150880	Single Family	1	0
94	SFR2004-00169	05/21/2004	1632150860	Single Family	1	0
95	SFR2004-00170	05/21/2004	1632150360	Single Family	1	0
96	SFR2004-00171	05/21/2004	1632150340	Single Family	1	0
97	SFR2004-00173	05/13/2004	1266840400	Single Family	1	0
98	SFR2004-00174	05/13/2004	1266840820	Single Family	1	0
99	SFR2004-00175	05/13/2004	1266840940	Single Family	1	0
100	SFR2004-00176	06/24/2004	1774964020	Single Family	1	0
101	SFR2004-00177	06/23/2004	1774964080	Single Family	1	0
102	SFR2004-00178	06/15/2004	1774964240	Single Family	1	0
103	SFR2004-00179	06/14/2004	1632150380	Single Family	1	0
104	SFR2004-00180	06/14/2004	1632150400	Single Family	1	0
105	SFR2004-00181	06/14/2004	1632150420	Single Family	1	0
106	SFR2004-00182	06/17/2004	1674290200	Single Family	1	0
107	SFR2004-00183	06/21/2004	1110226100	Single Family	1	0
108	SFR2004-00184	06/21/2004	1266841020	Single Family	1	0
109	SFR2004-00185	06/21/2004	1266841080	Single Family	1	0
110	SFR2004-00186	06/25/2004	0004400200	Single Family	1	0
111	SFR2004-00187	06/25/2004	1110226040	Single Family	1	0
112	SFR2004-00191	06/16/2004	1632150480	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
113	SFR2004-00192	06/16/2004	1632150500	Single Family	1	0
114	SFR2004-00199	06/30/2004	1632150900	Single Family	1	0
115	SFR2004-00200	06/30/2004	1632150940	Single Family	1	0
116	SFR2004-00203	06/29/2004	1632150220	Single Family	1	0
117	SFR2004-00204	06/29/2004	1632150240	Single Family	1	0
118	SFR2004-00205	06/21/2004	1266840160	Single Family	1	0
119	SFR2004-00212	06/16/2004	1632150640	Single Family	1	0
120	SFR2004-00213	06/16/2004	1632150660	Single Family	1	0
121	SFR2004-00223	06/21/2004	1266840800	Single Family	1	0
122	SFR2004-00233	06/23/2004	1098950140	Single Family	1	0
123	SFR2004-00238	06/23/2004	1098950420	Single Family	1	0
124	SFR2004-00239	06/23/2004	1098950400	Single Family	1	0
125	SFR2004-00240	06/23/2004	1098950440	Single Family	1	0
126	SFR2004-00241	06/23/2004	1098950160	Single Family	1	0
127	SFR2004-00242	06/23/2004	1098950380	Single Family	1	0
128	SFR2004-00243	06/23/2004	1098950520	Single Family	1	0
129	SFR2004-00244	06/23/2004	1098950180	Single Family	1	0
130	SFR2004-00245	06/23/2004	1098950200	Single Family	1	0
131	SFR2004-00246	06/23/2004	1098950460	Single Family	1	0
132	SFR2004-00247	06/23/2004	1098950500	Single Family	1	0
133	SFR2004-00248	06/23/2004	1098950480	Single Family	1	0
134	SFR2004-00250	06/23/2004	1098950360	Single Family	1	0
				Sub Total	152	21,657

Gromania UGA

Non-Residential

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
1	COM2003-00032	04/22/2004	1866200000	Other Commertial	0	15,000
2	COM2003-00197	05/25/2004	1060810000	Industrial	0	35,800
3	COM2003-00221	05/03/2004	1583510000	Industrial	0	30,000
4	COM2003-00222	05/11/2004	1583510000	Office	0	30,000
5	COM2003-00227	06/09/2004	1565110000	Industrial	0	6,062
6	COM2003-00228	06/09/2004	1565110000	Industrial	0	6,478
7	COM2003-00229	06/09/2004	1565110000	Industrial	0	13,068
8	COM2003-00230	06/09/2004	1565110000	Industrial	0	6,456
9	COM2003-00273	05/03/2004	1061200000	Commercial Retail	0	2,816
10	COM2003-00274	05/03/2004	1061200000	Commercial Retail	0	72
11	COM2003-00275	05/12/2004	1867870000	Office	0	98,233
12	COM2003-00276	06/25/2004	1867870000	Institutional	0	35,000
13	COM2003-00374	05/18/2004	1065290040	Office	0	3,100
14	COM2003-00407	05/06/2004	1077150000	Institutional	0	2,872
15	COM2003-00433	05/17/2004	1462230000	Institutional	0	30,000
16	COM2004-00104	05/26/2004	1586050000	Commercial Retail	0	6,000
17	COM2004-00108	06/04/2004	1873290000	Institutional	0	10,000
18	SFR2004-00823	06/11/2004	1555370240	Office	0	5,000

Residential

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
1	MFR2003-00019	06/23/2004	1178950710	Multi-Family	8	0
2	MFR2003-00020	06/23/2004	1178950710	Multi-Family	8	0
3	MFR2003-00021	06/23/2004	1178950710	Multi-Family	8	0
4	MFR2004-00018	06/10/2004	1079750000	Multi-Family	5	0
5	MFR2004-00025	05/14/2004	1079750000	Multi-Family	8	0
6	MFR2004-00026	06/22/2004	1864280000	Multi-Family	4	0
7	MFR2004-00030	06/15/2004	1079750000	Multi-Family	5	0
8	MOH2003-00195	04/05/2004	1174700000	Mobile Home	1	0
9	MOH2004-00019	04/01/2004	1181050880	Mobile Home	1	0
10	SFR2003-00649	04/12/2004	0974020000	Single Family	1	0
11	SFR2003-02106	05/25/2004	1820560200	Single Family	1	0
12	SFR2003-02142	04/26/2004	1820560540	Single Family	1	0
13	SFR2004-00130	05/24/2004	1856670500	Single Family	1	0
14	SFR2004-00134	05/24/2004	1856670020	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
15	SFR2004-00212	04/01/2004	1011930720	Single Family	1	0
16	SFR2004-00213	04/01/2004	1011930700	Single Family	1	0
17	SFR2004-00251	04/02/2004	0976610860	Single Family	1	0
18	SFR2004-00255	04/01/2004	1817820380	Single Family	1	0
19	SFR2004-00256	04/13/2004	1182549680	Single Family	1	0
20	SFR2004-00271	04/08/2004	1178940100	Single Family	1	0
21	SFR2004-00276	04/13/2004	1837080880	Single Family	1	0
22	SFR2004-00280	04/06/2004	1873300620	Single Family	1	0
23	SFR2004-00309	04/09/2004	1820590760	Single Family	1	0
24	SFR2004-00311	04/08/2004	1178940240	Single Family	1	0
25	SFR2004-00312	04/05/2004	0973310520	Single Family	1	0
26	SFR2004-00318	04/05/2004	1837080080	Single Family	1	0
27	SFR2004-00319	04/07/2004	1855750880	Single Family	1	0
28	SFR2004-00323	04/05/2004	0973310120	Single Family	1	0
29	SFR2004-00324	04/05/2004	0973310640	Single Family	1	0
30	SFR2004-00326	04/01/2004	1588661320	Single Family	1	0
31	SFR2004-00327	04/08/2004	1855752080	Single Family	1	0
32	SFR2004-00329	04/05/2004	1588661020	Single Family	1	0
33	SFR2004-00333	06/03/2004	1877860060	Single Family	1	0
34	SFR2004-00334	04/02/2004	1497570820	Single Family	1	0
35	SFR2004-00335	04/27/2004	1886670000	Single Family	1	0
36	SFR2004-00336	04/02/2004	1998480260	Single Family	1	0
37	SFR2004-00337	04/02/2004	1054091120	Single Family	1	0
38	SFR2004-00338	04/02/2004	0973100060	Single Family	1	0
39	SFR2004-00339	04/02/2004	0973100340	Single Family	1	0
40	SFR2004-00343	04/02/2004	0976610140	Single Family	1	0
41	SFR2004-00345	04/02/2004	0976611020	Single Family	1	0
42	SFR2004-00346	04/02/2004	0976611020	Single Family	1	0
43	SFR2004-00347	04/02/2004	0976611040	Single Family	1	0
44	SFR2004-00349	04/02/2004	1721260400	Single Family	1	0
45	SFR2004-00350	04/02/2004	1453720820	Single Family	1	0
46	SFR2004-00353	04/02/2004	1588660340	Single Family	1	0
47	SFR2004-00354	04/15/2004	1588660100	Single Family	1	0
48	SFR2004-00356	04/15/2004	1883220740	Single Family	1	0
49	SFR2004-00357	04/12/2004	1721980040	Single Family	1	0
50	SFR2004-00359	04/12/2004	1181043640	Single Family	1	0
51	SFR2004-00361	06/28/2004	1178977220	Single Family	1	0
52	SFR2004-00362	06/28/2004	1178977240	Single Family	1	0
53	SFR2004-00363	06/28/2004	1178977260	Single Family	1	0
54	SFR2004-00364	06/28/2004	1178977280	Single Family	1	0
55	SFR2004-00365	04/07/2004	1588660640	Single Family	1	0
56	SFR2004-00366	04/07/2004	1588660160	Single Family	1	0
57	SFR2004-00367	04/07/2004	1588661540	Single Family	1	0
58	SFR2004-00368	04/15/2004	1852090000	Single Family	1	0
59	SFR2004-00370	04/22/2004	1085380780	Single Family	1	0
60	SFR2004-00371	04/16/2004	0973502240	Single Family	1	0
61	SFR2004-00373	04/16/2004	0973500660	Single Family	1	0
62	SFR2004-00374	04/16/2004	0973502000	Single Family	1	0
63	SFR2004-00375	04/16/2004	0973502380	Single Family	1	0
64	SFR2004-00376	04/16/2004	0973501040	Single Family	1	0
65	SFR2004-00377	05/12/2004	1820580760	Single Family	1	0
66	SFR2004-00380	05/26/2004	1873300020	Single Family	1	0
67	SFR2004-00384	04/07/2004	1855753160	Single Family	1	0
68	SFR2004-00385	04/08/2004	0973100480	Single Family	1	0
69	SFR2004-00386	04/08/2004	0973100520	Single Family	1	0
70	SFR2004-00387	04/19/2004	0973100540	Single Family	1	0
71	SFR2004-00388	04/06/2004	1873300220	Single Family	1	0
72	SFR2004-00390	04/05/2004	1056128380	Single Family	1	0
73	SFR2004-00391	05/03/2004	1873300580	Single Family	1	0
74	SFR2004-00392	04/28/2004	1178940180	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
75	SFR2004-00397	04/07/2004	1588661060	Single Family	1	0
76	SFR2004-00398	04/14/2004	1182641760	Single Family	1	0
77	SFR2004-00399	04/07/2004	1588660760	Single Family	1	0
78	SFR2004-00403	05/17/2004	1873300300	Single Family	1	0
79	SFR2004-00404	04/07/2004	1593732760	Single Family	1	0
80	SFR2004-00405	04/07/2004	1593732080	Single Family	1	0
81	SFR2004-00406	04/07/2004	1593732800	Single Family	1	0
82	SFR2004-00407	04/07/2004	1593732880	Single Family	1	0
83	SFR2004-00408	04/07/2004	1593733480	Single Family	1	0
84	SFR2004-00409	04/07/2004	1593732560	Single Family	1	0
85	SFR2004-00410	04/12/2004	1721980180	Single Family	1	0
86	SFR2004-00411	05/27/2004	1181043680	Single Family	1	0
87	SFR2004-00412	04/28/2004	1181043620	Single Family	1	0
88	SFR2004-00413	05/27/2004	1181043660	Single Family	1	0
89	SFR2004-00414	05/11/2004	1178940120	Single Family	1	0
90	SFR2004-00416	04/15/2004	1593733460	Single Family	1	0
91	SFR2004-00418	04/30/2004	1056128400	Single Family	1	0
92	SFR2004-00419	04/15/2004	1593732740	Single Family	1	0
93	SFR2004-00420	04/15/2004	1593733380	Single Family	1	0
94	SFR2004-00421	04/15/2004	1593731980	Single Family	1	0
95	SFR2004-00423	04/23/2004	1593730040	Single Family	1	0
96	SFR2004-00424	04/15/2004	1593732220	Single Family	1	0
97	SFR2004-00425	04/27/2004	1644990740	Single Family	1	0
98	SFR2004-00426	04/12/2004	1644990660	Single Family	1	0
99	SFR2004-00427	04/16/2004	0976610080	Single Family	1	0
100	SFR2004-00428	04/23/2004	1855752820	Single Family	1	0
101	SFR2004-00429	04/15/2004	1593733340	Single Family	1	0
102	SFR2004-00432	04/19/2004	1588660500	Single Family	1	0
103	SFR2004-00433	04/19/2004	1588660800	Single Family	1	0
104	SFR2004-00435	04/16/2004	1593733680	Single Family	1	0
105	SFR2004-00436	04/16/2004	1593731120	Single Family	1	0
106	SFR2004-00439	04/19/2004	1588660180	Single Family	1	0
107	SFR2004-00440	04/19/2004	1855752090	Single Family	1	0
108	SFR2004-00441	04/23/2004	1891800020	Single Family	1	0
109	SFR2004-00442	04/23/2004	1891800040	Single Family	1	0
110	SFR2004-00443	04/23/2004	1891800060	Single Family	1	0
111	SFR2004-00444	04/23/2004	1891800080	Single Family	1	0
112	SFR2004-00445	04/23/2004	1891800100	Single Family	1	0
113	SFR2004-00446	04/23/2004	1891800120	Single Family	1	0
114	SFR2004-00447	04/23/2004	1891800140	Single Family	1	0
115	SFR2004-00448	04/23/2004	1891800160	Single Family	1	0
116	SFR2004-00450	05/24/2004	1891800420	Single Family	1	0
117	SFR2004-00451	05/24/2004	1891800440	Single Family	1	0
118	SFR2004-00452	05/24/2004	1891800460	Single Family	1	0
119	SFR2004-00453	05/24/2004	1891800480	Single Family	1	0
120	SFR2004-00454	05/24/2004	1891800500	Single Family	1	0
121	SFR2004-00455	05/24/2004	1891800520	Single Family	1	0
122	SFR2004-00456	05/24/2004	1891800540	Single Family	1	0
123	SFR2004-00457	05/24/2004	1891800560	Single Family	1	0
124	SFR2004-00458	04/22/2004	1593733500	Single Family	1	0
125	SFR2004-00459	04/19/2004	1588660240	Single Family	1	0
126	SFR2004-00460	04/19/2004	1588661500	Single Family	1	0
127	SFR2004-00461	04/19/2004	1721930060	Single Family	1	0
128	SFR2004-00463	05/11/2004	1860190380	Single Family	1	0
129	SFR2004-00465	05/03/2004	1555370700	Single Family	1	0
130	SFR2004-00467	05/10/2004	1447450000	Single Family	1	0
131	SFR2004-00468	04/16/2004	0976611060	Single Family	1	0
132	SFR2004-00469	04/16/2004	0976611120	Single Family	1	0
133	SFR2004-00470	04/16/2004	1453720940	Single Family	1	0
134	SFR2004-00471	04/16/2004	0976611100	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
135	SFR2004-00472	04/16/2004	1453720660	Single Family	1	0
136	SFR2004-00473	04/22/2004	1588660360	Single Family	1	0
137	SFR2004-00474	04/22/2004	1593733420	Single Family	1	0
138	SFR2004-00478	05/04/2004	1609910040	Single Family	1	0
139	SFR2004-00482	04/16/2004	0973500700	Single Family	1	0
140	SFR2004-00483	04/16/2004	0973500880	Single Family	1	0
141	SFR2004-00484	04/16/2004	0973500980	Single Family	1	0
142	SFR2004-00485	04/16/2004	0973501920	Single Family	1	0
143	SFR2004-00486	04/16/2004	0973502360	Single Family	1	0
144	SFR2004-00487	05/25/2004	0973502940	Single Family	1	0
145	SFR2004-00488	05/25/2004	0973502960	Single Family	1	0
146	SFR2004-00489	05/25/2004	0973502980	Single Family	1	0
147	SFR2004-00490	05/25/2004	0973503000	Single Family	1	0
148	SFR2004-00491	04/21/2004	0973503100	Single Family	1	0
149	SFR2004-00492	04/21/2004	0973503120	Single Family	1	0
150	SFR2004-00493	04/21/2004	0973503140	Single Family	1	0
151	SFR2004-00494	04/27/2004	0973503160	Single Family	1	0
152	SFR2004-00495	04/27/2004	0973503180	Single Family	1	0
153	SFR2004-00496	04/27/2004	0973503200	Single Family	1	0
154	SFR2004-00497	04/27/2004	0973503220	Single Family	1	0
155	SFR2004-00498	04/16/2004	0976611080	Single Family	1	0
156	SFR2004-00500	04/22/2004	0973100380	Single Family	1	0
157	SFR2004-00501	04/22/2004	0973100500	Single Family	1	0
158	SFR2004-00502	04/22/2004	0973100560	Single Family	1	0
159	SFR2004-00503	04/22/2004	0973100720	Single Family	1	0
160	SFR2004-00504	04/22/2004	1998480020	Single Family	1	0
161	SFR2004-00505	04/22/2004	1998480700	Single Family	1	0
162	SFR2004-00506	04/22/2004	1054090340	Single Family	1	0
163	SFR2004-00507	04/22/2004	1054090420	Single Family	1	0
164	SFR2004-00508	04/22/2004	1054090660	Single Family	1	0
165	SFR2004-00509	05/04/2004	1055060080	Single Family	1	0
166	SFR2004-00511	05/05/2004	1539370060	Single Family	1	0
167	SFR2004-00513	04/23/2004	1588660040	Single Family	1	0
168	SFR2004-00514	04/23/2004	1593730320	Single Family	1	0
169	SFR2004-00515	04/16/2004	1593730100	Single Family	1	0
170	SFR2004-00516	04/16/2004	1593732240	Single Family	1	0
171	SFR2004-00517	05/04/2004	1054090140	Single Family	1	0
172	SFR2004-00519	04/19/2004	1588660700	Single Family	1	0
173	SFR2004-00520	05/03/2004	1555371480	Single Family	1	0
174	SFR2004-00521	04/19/2004	1588660740	Single Family	1	0
175	SFR2004-00522	04/19/2004	1593733740	Single Family	1	0
176	SFR2004-00523	04/19/2004	1855752860	Single Family	1	0
177	SFR2004-00524	05/03/2004	1555371560	Single Family	1	0
178	SFR2004-00525	05/05/2004	1445060780	Single Family	1	0
179	SFR2004-00526	05/05/2004	1445060800	Single Family	1	0
180	SFR2004-00527	05/04/2004	1445060620	Single Family	1	0
181	SFR2004-00528	05/04/2004	1445060640	Single Family	1	0
182	SFR2004-00529	05/04/2004	1445060660	Single Family	1	0
183	SFR2004-00530	05/04/2004	1445060680	Single Family	1	0
184	SFR2004-00531	05/05/2004	1445060540	Single Family	1	0
185	SFR2004-00532	05/05/2004	1445060560	Single Family	1	0
186	SFR2004-00533	05/05/2004	1445060580	Single Family	1	0
187	SFR2004-00534	05/05/2004	1445060600	Single Family	1	0
188	SFR2004-00536	05/05/2004	1453720980	Single Family	1	0
189	SFR2004-00537	04/27/2004	0976610180	Single Family	1	0
190	SFR2004-00538	05/05/2004	0976610220	Single Family	1	0
191	SFR2004-00539	04/27/2004	0976610100	Single Family	1	0
192	SFR2004-00540	05/05/2004	0976610980	Single Family	1	0
193	SFR2004-00541	05/11/2004	0966230800	Single Family	1	0
194	SFR2004-00542	04/27/2004	1644990680	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
195	SFR2004-00545	04/27/2004	1588660560	Single Family	1	0
196	SFR2004-00546	04/27/2004	1855753060	Single Family	1	0
197	SFR2004-00550	05/04/2004	0973500100	Single Family	1	0
198	SFR2004-00551	04/27/2004	0973500120	Single Family	1	0
199	SFR2004-00552	04/27/2004	0973501140	Single Family	1	0
200	SFR2004-00553	05/04/2004	0973501200	Single Family	1	0
201	SFR2004-00554	05/04/2004	0973502180	Single Family	1	0
202	SFR2004-00556	05/04/2004	1856670480	Single Family	1	0
203	SFR2004-00557	05/04/2004	1856670620	Single Family	1	0
204	SFR2004-00558	04/26/2004	1849500440	Single Family	1	0
205	SFR2004-00559	05/10/2004	0973310600	Single Family	1	0
206	SFR2004-00560	05/17/2004	1873300080	Single Family	1	0
207	SFR2004-00563	05/12/2004	1555371540	Single Family	1	0
208	SFR2004-00566	04/26/2004	1593731040	Single Family	1	0
209	SFR2004-00567	04/22/2004	1593732320	Single Family	1	0
210	SFR2004-00568	04/22/2004	1593732280	Single Family	1	0
211	SFR2004-00569	04/22/2004	1593732840	Single Family	1	0
212	SFR2004-00570	04/22/2004	1593733280	Single Family	1	0
213	SFR2004-00571	04/22/2004	1593730120	Single Family	1	0
214	SFR2004-00573	05/10/2004	0973310180	Single Family	1	0
215	SFR2004-00575	05/05/2004	1900020640	Single Family	1	0
216	SFR2004-00576	05/05/2004	1900020760	Single Family	1	0
217	SFR2004-00577	05/05/2004	1900020720	Single Family	1	0
218	SFR2004-00578	05/05/2004	1900020960	Single Family	1	0
219	SFR2004-00580	05/18/2004	1856670260	Single Family	1	0
220	SFR2004-00583	06/07/2004	1856670300	Single Family	1	0
221	SFR2004-00586	04/30/2004	1721980200	Single Family	1	0
222	SFR2004-00587	04/30/2004	1721980080	Single Family	1	0
223	SFR2004-00589	05/25/2004	1453720780	Single Family	1	0
224	SFR2004-00590	05/18/2004	1453720640	Single Family	1	0
225	SFR2004-00591	05/25/2004	1453720800	Single Family	1	0
226	SFR2004-00592	05/05/2004	0976610200	Single Family	1	0
227	SFR2004-00593	05/07/2004	1882570050	Single Family	1	0
228	SFR2004-00594	04/22/2004	1593731100	Single Family	1	0
229	SFR2004-00595	04/29/2004	1593733240	Single Family	1	0
230	SFR2004-00597	05/21/2004	0973100200	Single Family	1	0
231	SFR2004-00598	05/21/2004	0973100040	Single Family	1	0
232	SFR2004-00599	05/21/2004	0973100160	Single Family	1	0
233	SFR2004-00600	05/14/2004	1054090720	Single Family	1	0
234	SFR2004-00601	05/04/2004	1054090840	Single Family	1	0
235	SFR2004-00602	05/14/2004	1054090200	Single Family	1	0
236	SFR2004-00603	05/14/2004	1054091100	Single Family	1	0
237	SFR2004-00604	05/04/2004	1054090280	Single Family	1	0
238	SFR2004-00605	05/21/2004	1497570840	Single Family	1	0
239	SFR2004-00607	05/12/2004	1453720620	Single Family	1	0
240	SFR2004-00608	06/04/2004	1468150700	Single Family	1	0
241	SFR2004-00609	05/04/2004	0973500020	Single Family	1	0
242	SFR2004-00610	05/11/2004	0973501100	Single Family	1	0
243	SFR2004-00611	05/11/2004	0973501120	Single Family	1	0
244	SFR2004-00612	05/11/2004	0973501940	Single Family	1	0
245	SFR2004-00613	05/11/2004	0973502080	Single Family	1	0
246	SFR2004-00615	05/11/2004	0973501180	Single Family	1	0
247	SFR2004-00616	05/10/2004	0973310580	Single Family	1	0
248	SFR2004-00619	05/04/2004	1820590660	Single Family	1	0
249	SFR2004-00620	05/04/2004	1820590700	Single Family	1	0
250	SFR2004-00621	05/04/2004	1820590540	Single Family	1	0
251	SFR2004-00625	05/04/2004	1593733720	Single Family	1	0
252	SFR2004-00627	05/04/2004	1593733640	Single Family	1	0
253	SFR2004-00628	05/11/2004	1593730220	Single Family	1	0
254	SFR2004-00629	05/04/2004	1593733400	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
255	SFR2004-00630	05/04/2004	1593733320	Single Family	1	0
256	SFR2004-00631	05/11/2004	1593732040	Single Family	1	0
257	SFR2004-00633	04/29/2004	1593732340	Single Family	1	0
258	SFR2004-00634	05/17/2004	1178977180	Single Family	1	0
259	SFR2004-00635	05/17/2004	1178977120	Single Family	1	0
260	SFR2004-00636	05/24/2004	0988242120	Single Family	1	0
261	SFR2004-00637	05/17/2004	1178977140	Single Family	1	0
262	SFR2004-00638	05/17/2004	1178977160	Single Family	1	0
263	SFR2004-00639	05/24/2004	0988242100	Single Family	1	0
264	SFR2004-00640	05/17/2004	1178977200	Single Family	1	0
265	SFR2004-00641	06/07/2004	1856670320	Single Family	1	0
266	SFR2004-00643	06/07/2004	1856670280	Single Family	1	0
267	SFR2004-00645	05/17/2004	1856670240	Single Family	1	0
268	SFR2004-00646	06/14/2004	1177685240	Single Family	1	0
269	SFR2004-00649	05/11/2004	1593731080	Single Family	1	0
270	SFR2004-00650	05/11/2004	1593731140	Single Family	1	0
271	SFR2004-00651	05/18/2004	1593732160	Single Family	1	0
272	SFR2004-00652	05/18/2004	1593730180	Single Family	1	0
273	SFR2004-00653	05/18/2004	1593733520	Single Family	1	0
274	SFR2004-00654	05/18/2004	1593730160	Single Family	1	0
275	SFR2004-00655	05/05/2004	1453720960	Single Family	1	0
276	SFR2004-00659	05/04/2004	1593730080	Single Family	1	0
277	SFR2004-00660	05/12/2004	0976610040	Single Family	1	0
278	SFR2004-00661	05/04/2004	1593733620	Single Family	1	0
279	SFR2004-00662	05/12/2004	0976610120	Single Family	1	0
280	SFR2004-00663	05/03/2004	1855753080	Single Family	1	0
281	SFR2004-00665	05/03/2004	1855753000	Single Family	1	0
282	SFR2004-00670	05/24/2004	1593730640	Single Family	1	0
283	SFR2004-00671	05/24/2004	1593730140	Single Family	1	0
284	SFR2004-00672	05/20/2004	1593733560	Single Family	1	0
285	SFR2004-00673	05/20/2004	1593732600	Single Family	1	0
286	SFR2004-00674	05/20/2004	1593733800	Single Family	1	0
287	SFR2004-00675	05/20/2004	1593732520	Single Family	1	0
288	SFR2004-00676	05/24/2004	1837080280	Single Family	1	0
289	SFR2004-00677	05/12/2004	1873300260	Single Family	1	0
290	SFR2004-00679	05/04/2004	0973501160	Single Family	1	0
291	SFR2004-00680	05/04/2004	0973501880	Single Family	1	0
292	SFR2004-00681	05/04/2004	0973502100	Single Family	1	0
293	SFR2004-00682	05/04/2004	0973502160	Single Family	1	0
294	SFR2004-00683	06/01/2004	1849630360	Single Family	1	0
295	SFR2004-00684	05/18/2004	1849630260	Single Family	1	0
296	SFR2004-00687	06/29/2004	1849630300	Single Family	1	0
297	SFR2004-00690	05/18/2004	1849630220	Single Family	1	0
298	SFR2004-00691	06/01/2004	1849630380	Single Family	1	0
299	SFR2004-00693	05/12/2004	1593732660	Single Family	1	0
300	SFR2004-00694	05/20/2004	1054090400	Single Family	1	0
301	SFR2004-00695	05/14/2004	1054090700	Single Family	1	0
302	SFR2004-00696	05/14/2004	1054091080	Single Family	1	0
303	SFR2004-00697	05/14/2004	1054091180	Single Family	1	0
304	SFR2004-00698	06/18/2004	0973100020	Single Family	1	0
305	SFR2004-00699	05/21/2004	0973100120	Single Family	1	0
306	SFR2004-00700	05/21/2004	0973100180	Single Family	1	0
307	SFR2004-00701	05/21/2004	0973100420	Single Family	1	0
308	SFR2004-00702	06/18/2004	0973100740	Single Family	1	0
309	SFR2004-00703	06/18/2004	0973100800	Single Family	1	0
310	SFR2004-00704	06/18/2004	1998480400	Single Family	1	0
311	SFR2004-00705	06/16/2004	0982835060	Single Family	1	0
312	SFR2004-00706	05/21/2004	1998480440	Single Family	1	0
313	SFR2004-00707	05/21/2004	1569440220	Single Family	1	0
314	SFR2004-00713	05/11/2004	1593732260	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
315	SFR2004-00714	05/11/2004	1593733220	Single Family	1	0
316	SFR2004-00715	05/11/2004	1593733660	Single Family	1	0
317	SFR2004-00716	05/11/2004	1593733700	Single Family	1	0
318	SFR2004-00717	05/11/2004	1588661160	Single Family	1	0
319	SFR2004-00718	04/29/2004	1058710020	Single Family	1	0
320	SFR2004-00720	04/29/2004	1058710060	Single Family	1	0
321	SFR2004-00721	05/17/2004	1856670540	Single Family	1	0
322	SFR2004-00722	06/29/2004	1849630280	Single Family	1	0
323	SFR2004-00723	05/17/2004	1555371500	Single Family	1	0
324	SFR2004-00725	05/20/2004	0973310060	Single Family	1	0
325	SFR2004-00731	05/25/2004	0973502040	Single Family	1	0
326	SFR2004-00732	05/25/2004	0973501960	Single Family	1	0
327	SFR2004-00733	05/25/2004	0973501580	Single Family	1	0
328	SFR2004-00734	05/25/2004	0973500640	Single Family	1	0
329	SFR2004-00735	05/25/2004	0973500400	Single Family	1	0
330	SFR2004-00737	05/10/2004	1593733820	Single Family	1	0
331	SFR2004-00738	05/10/2004	1593732780	Single Family	1	0
332	SFR2004-00739	05/10/2004	1593733580	Single Family	1	0
333	SFR2004-00741	05/10/2004	1588660580	Single Family	1	0
334	SFR2004-00745	05/24/2004	1593733440	Single Family	1	0
335	SFR2004-00746	05/24/2004	1593730240	Single Family	1	0
336	SFR2004-00747	05/27/2004	1593733360	Single Family	1	0
337	SFR2004-00748	05/27/2004	1593732400	Single Family	1	0
338	SFR2004-00752	06/11/2004	1555370360	Single Family	1	0
339	SFR2004-00753	05/28/2004	1555371160	Single Family	1	0
340	SFR2004-00754	06/24/2004	1054090360	Single Family	1	0
341	SFR2004-00755	05/21/2004	1054090480	Single Family	1	0
342	SFR2004-00758	06/21/2004	1877860020	Single Family	1	0
343	SFR2004-00761	05/10/2004	1593733260	Single Family	1	0
344	SFR2004-00762	05/10/2004	1588660060	Single Family	1	0
345	SFR2004-00763	05/10/2004	1588660320	Single Family	1	0
346	SFR2004-00764	06/03/2004	1177685940	Single Family	1	0
347	SFR2004-00768	05/25/2004	0976610880	Single Family	1	0
348	SFR2004-00770	05/25/2004	0976610160	Single Family	1	0
349	SFR2004-00775	05/20/2004	1544590020	Single Family	1	0
350	SFR2004-00776	05/20/2004	1544590040	Single Family	1	0
351	SFR2004-00777	05/20/2004	1544590060	Single Family	1	0
352	SFR2004-00778	05/20/2004	1544590080	Single Family	1	0
353	SFR2004-00779	05/20/2004	1544590100	Single Family	1	0
354	SFR2004-00780	05/14/2004	1873300600	Single Family	1	0
355	SFR2004-00784	06/04/2004	1837080160	Single Family	1	0
356	SFR2004-00786	05/20/2004	0973310700	Single Family	1	0
357	SFR2004-00788	06/21/2004	1877860040	Single Family	1	0
358	SFR2004-00793	05/25/2004	0973500040	Single Family	1	0
359	SFR2004-00794	05/25/2004	0973500140	Single Family	1	0
360	SFR2004-00796	05/25/2004	0973501540	Single Family	1	0
361	SFR2004-00797	05/25/2004	0973503240	Single Family	1	0
362	SFR2004-00798	05/25/2004	0973503260	Single Family	1	0
363	SFR2004-00799	05/25/2004	0973503280	Single Family	1	0
364	SFR2004-00800	05/25/2004	0973503300	Single Family	1	0
365	SFR2004-00801	05/25/2004	0973503320	Single Family	1	0
366	SFR2004-00802	05/25/2004	0973503340	Single Family	1	0
367	SFR2004-00803	05/25/2004	0973503360	Single Family	1	0
368	SFR2004-00804	05/25/2004	0973503380	Single Family	1	0
369	SFR2004-00805	05/25/2004	0973503400	Single Family	1	0
370	SFR2004-00824	06/11/2004	1555370240	Single Family	1	0
371	SFR2004-00826	05/20/2004	1593732500	Single Family	1	0
372	SFR2004-00827	06/04/2004	1593732120	Single Family	1	0
373	SFR2004-00828	06/04/2004	1593733080	Single Family	1	0
374	SFR2004-00829	06/04/2004	1593730200	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
375	SFR2004-00836	05/25/2004	1644990700	Single Family	1	0
376	SFR2004-00838	05/21/2004	1445061220	Single Family	1	0
377	SFR2004-00839	05/21/2004	1445061240	Single Family	1	0
378	SFR2004-00840	05/21/2004	1445061140	Single Family	1	0
379	SFR2004-00841	05/21/2004	1445061160	Single Family	1	0
380	SFR2004-00842	05/21/2004	1445060700	Single Family	1	0
381	SFR2004-00843	05/21/2004	1445060720	Single Family	1	0
382	SFR2004-00844	05/21/2004	1445060740	Single Family	1	0
383	SFR2004-00845	05/21/2004	1445060760	Single Family	1	0
384	SFR2004-00850	06/17/2004	1873300160	Single Family	1	0
385	SFR2004-00851	06/07/2004	1086602940	Single Family	1	0
386	SFR2004-00853	06/24/2004	1882570000	Single Family	1	0
387	SFR2004-00855	06/24/2004	1882570150	Single Family	1	0
388	SFR2004-00856	05/20/2004	1593733840	Single Family	1	0
389	SFR2004-00857	06/09/2004	1555371520	Single Family	1	0
390	SFR2004-00895	06/23/2004	1856670640	Single Family	1	0
391	SFR2004-00896	06/23/2004	1856670660	Single Family	1	0
392	SFR2004-00897	06/23/2004	1856670440	Single Family	1	0
393	SFR2004-00898	06/23/2004	1856670460	Single Family	1	0
394	SFR2004-00900	05/24/2004	0973310100	Single Family	1	0
395	SFR2004-00901	06/22/2004	0973310420	Single Family	1	0
396	SFR2004-00902	06/22/2004	1992370050	Single Family	1	0
397	SFR2004-00913	05/28/2004	1858381980	Single Family	1	0
398	SFR2004-00914	05/28/2004	1858382000	Single Family	1	0
399	SFR2004-00917	06/15/2004	1644990720	Single Family	1	0
400	SFR2004-00919	06/04/2004	0973500280	Single Family	1	0
401	SFR2004-00920	06/04/2004	0973500300	Single Family	1	0
402	SFR2004-00921	06/04/2004	0973500320	Single Family	1	0
403	SFR2004-00922	06/04/2004	0973500360	Single Family	1	0
404	SFR2004-00926	06/04/2004	0973501600	Single Family	1	0
405	SFR2004-00927	06/04/2004	0973501620	Single Family	1	0
406	SFR2004-00928	06/04/2004	0973501640	Single Family	1	0
407	SFR2004-00930	06/04/2004	0973501700	Single Family	1	0
408	SFR2004-00931	06/04/2004	0973501720	Single Family	1	0
409	SFR2004-00932	06/04/2004	0973501820	Single Family	1	0
410	SFR2004-00933	06/04/2004	0973501840	Single Family	1	0
411	SFR2004-00935	06/04/2004	0973501900	Single Family	1	0
412	SFR2004-00942	06/18/2004	1889410160	Single Family	1	0
413	SFR2004-00945	05/27/2004	1855753500	Single Family	1	0
414	SFR2004-00947	06/14/2004	1076626340	Single Family	1	0
415	SFR2004-00979	06/09/2004	0973310720	Single Family	1	0
416	SFR2004-00980	06/09/2004	0973310200	Single Family	1	0
417	SFR2004-00981	06/08/2004	1058710100	Single Family	1	0
418	SFR2004-01028	06/08/2004	1058710080	Single Family	1	0
419	SFR2004-01059	06/10/2004	1445060520	Single Family	1	0
420	SFR2004-01061	06/10/2004	1445060500	Single Family	1	0
421	SFR2004-01062	06/10/2004	1445060480	Single Family	1	0
422	SFR2004-01063	06/10/2004	1445060460	Single Family	1	0
423	SFR2004-01064	06/10/2004	1445061100	Single Family	1	0
424	SFR2004-01065	06/10/2004	1445061120	Single Family	1	0
425	SFR2004-01066	06/10/2004	1445061020	Single Family	1	0
426	SFR2004-01067	06/10/2004	1445061040	Single Family	1	0
427	SFR2004-01068	06/10/2004	1445060900	Single Family	1	0
428	SFR2004-01069	06/10/2004	1445060920	Single Family	1	0
429	SFR2004-01140	06/07/2004	0976611000	Single Family	1	0
430	SFR2004-01141	06/14/2004	1593733760	Single Family	1	0
431	SFR2004-01142	06/14/2004	1593733120	Single Family	1	0
432	SFR2004-01143	06/14/2004	1593733160	Single Family	1	0
433	SFR2004-01145	06/28/2004	1721980220	Single Family	1	0
434	SFR2004-01169	06/25/2004	0973310400	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
435	SFR2004-01170	06/11/2004	1593732060	Single Family	1	0
436	SFR2004-01171	06/11/2004	1593732100	Single Family	1	0
437	SFR2004-01174	06/11/2004	1593732180	Single Family	1	0
438	SFR2004-01175	06/11/2004	1593732020	Single Family	1	0
439	SFR2004-01176	06/11/2004	1593732900	Single Family	1	0
440	SFR2004-01177	06/21/2004	1593732460	Single Family	1	0
441	SFR2004-01211	06/11/2004	1181400880	Single Family	1	0
442	SFR2004-01213	06/16/2004	1593733140	Single Family	1	0
443	SFR2004-01214	06/16/2004	1593733100	Single Family	1	0
444	SFR2004-01216	06/16/2004	1593733540	Single Family	1	0
445	SFR2004-01253	06/16/2004	1593733300	Single Family	1	0
446	SFR2004-01294	06/22/2004	1593730060	Single Family	1	0
447	SFR2004-01299	06/29/2004	1593733200	Single Family	1	0
448	SFR2004-01300	06/30/2004	1593733600	Single Family	1	0
				Sub Total	487	335,957

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
Waterburg						
Non-Residential						
1	2005-362	03/21/2005	1327921180	Commercial Retail	0	900
2	2005-366	03/21/2005	1327920560	Commercial Retail	0	1,100
3	2005-370	03/21/2005	1327921160	Commercial Retail	0	1,300
4	2005-374	03/21/2005	1327920500	Commercial Retail	0	500
5	2005-378	03/16/2005	1327920620	Commercial Retail	0	3,456
6	2005-382	03/16/2005	1327921140	Office	0	1,234
7	2005-39	03/02/2005	1303610200	Office	0	2,345
8	2005-462	03/29/2005	1327920660	Office	0	4,321
9	2005-466	03/29/2005	1327921100	Institutional	0	10,000
10	2005-50	03/01/2005	1230060920	Institutional	0	14,600
11	2005-51	03/11/2005	1230020540	Industrial	0	20,080
12	2005-65	03/03/2005	1327920700	Other Commertial	0	4,000
13	2005-66	03/08/2005	1305561880	Other Commertial	0	3,600
14	2005-67	03/28/2005	1283580660	Other Commertial	0	1,600
Residential						
1	2004-1353	02/07/2005	1300710100	Single Family	1	0
2	2004-1366	01/18/2005	1301150460	Single Family	1	0
3	2004-1409	01/27/2005	1283580040	Single Family	1	0
4	2004-1437	03/01/2005	1283580060	Single Family	1	0
5	2004-1438	01/20/2005	1302740380	Single Family	1	0
6	2004-1440	01/13/2005	1302740060	Single Family	1	0
7	2004-1441	01/19/2005	1303610180	Single Family	1	0
8	2004-1443	01/14/2005	1324140180	Single Family	1	0
9	2004-1444	01/28/2005	1230060400	Single Family	1	0
10	2004-1445	01/12/2005	1230060600	Single Family	1	0
11	2004-1446	01/28/2005	1230060420	Single Family	1	0
12	2004-1452	01/31/2005	1300710220	Single Family	1	0
13	2004-1455	01/13/2005	1283580220	Single Family	1	0
14	2004-1477	02/04/2005	1283580480	Single Family	1	0
15	2004-1517	02/03/2005	1303610300	Single Family	1	0
16	2004-1550	01/13/2005	1230060300	Single Family	1	0
17	2004-1566	02/03/2005	1283580700	Single Family	1	0
18	2004-1571	02/18/2005	1327920960	Single Family	1	0
19	2004-1582	01/13/2005	1326080180	Single Family	1	0
20	2004-1583	01/06/2005	1300710660	Single Family	1	0
21	2004-1584	02/03/2005	1305560860	Single Family	1	0
22	2004-1585	01/27/2005	1230060280	Single Family	1	0
23	2004-1624	02/22/2005	1283580020	Single Family	1	0
24	2004-1641	02/03/2005	1283580460	Single Family	1	0
25	2004-1642	01/27/2005	1303610040	Single Family	1	0
26	2004-1643	02/03/2005	1303610020	Single Family	1	0
27	2004-1644	02/03/2005	1305560540	Single Family	1	0
28	2004-1656	02/24/2005	1283580500	Single Family	1	0
29	2004-1673	02/15/2005	1302740420	Single Family	1	0
30	2004-1680	03/02/2005	1303610340	Single Family	1	0
31	2004-1681	02/14/2005	1302740200	Single Family	1	0
32	2004-1694	02/24/2005	1283580620	Single Family	1	0
33	2004-1695	02/14/2005	1283580640	Single Family	1	0
34	2004-1701	03/02/2005	1283580960	Single Family	1	0
35	2004-1710	02/24/2005	1301151580	Single Family	1	0
36	2004-1719	02/22/2005	1230010240	Single Family	1	0
37	2004-1720	02/22/2005	1230010300	Single Family	1	0
38	2004-1724	02/23/2005	1303610220	Single Family	1	0
39	2004-1738	03/01/2005	1302740140	Single Family	1	0
40	2004-1743	02/22/2005	1283580580	Single Family	1	0
41	2004-1744	02/28/2005	1283580400	Single Family	1	0
42	2004-1745	02/28/2005	1283580100	Multi-Family	10	0
43	2004-1746	02/28/2005	1283580340	Multi-Family	4	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
44	2004-1752	02/24/2005	1283580600	Multi-Family	8	0
45	2004-1753	02/23/2005	1283580680	Multi-Family	2	0
46	2004-1755	02/28/2005	1303610280	Multi-Family	6	0
47	2005-123	02/04/2005	1327921220	Duplex	2	0
48	2005-127	02/04/2005	1327920540	Duplex	2	0
49	2005-155	02/14/2005	1303610060	Duplex	2	0
50	2005-159	03/02/2005	1303610080	Duplex	2	0
51	2005-163	03/30/2005	1230060840	Duplex	2	0
52	2005-174	02/14/2005	1303610420	Mobile Home	1	0
53	2005-198	03/01/2005	1301151500	Mobile Home	1	0
54	2005-202	02/18/2005	1301151360	Mobile Home	1	0
55	2005-21	03/09/2005	1283580960	Mobile Home	1	0
56	2005-213	03/08/2005	1230060900	Mobile Home	1	0
57	2005-217	02/16/2005	1327921080	Additions	1	0
58	2005-221	02/23/2005	1327920820	Additions	1	0
59	2005-225	02/16/2005	1327920640	Additions	1	0
60	2005-229	02/18/2005	1327920480	Additions	1	0
61	2005-233	02/16/2005	1327921240	Additions	1	0
62	2005-283	03/08/2005	1303610400	Mobile Home Park	1	0
63	2005-293	02/28/2005	1303610320	Mobile Home Park	1	0
64	2005-31	01/13/2005	1230060560	Mobile Home Park	1	0
65	2005-354	03/16/2005	1327921200	Mobile Home Park	1	0
66	2005-358	03/16/2005	1327920520	Mobile Home Park	1	0
67	2005-88	03/28/2005	1301151280	Single Family	1	0
68	2005-94	03/28/2005	1301151520	Single Family	1	0
69	2005-95	03/29/2005	1303610100	Single Family	1	0
				Sub Total	99	69,036
Waterburg UGA						
Non-Residential						
1	SubuMadeUp-017	01/01/2005	1244910000	Commercial Retail	0	1,200
Residential						
1	SFR2003-02130	04/16/2004	1304160150	Single Family	1	0
2	SFR2004-00060	04/27/2004	1302870000	Single Family	1	0
3	SFR2004-00302	04/16/2004	1304260050	Single Family	1	0
				Sub Total	3	1,200

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
Treehaven						
Non-Residential						
1	SFR2004-00543	04/26/2004	0668640000	Industrial	0	100,000
Residential						
1	SFR2004-00396	04/15/2004	0658600000	Single Family	1	0
2	SFR2004-00543	04/26/2004	0668640000	Single Family	1	0
3	SFR2004-00544	04/26/2004	0668600000	Single Family	1	0
4	SFR2004-01034	06/15/2004	0650960000	Single Family	1	0
				Sub Total	4	100,000
Treehaven UGA						
Non-Residential						
1	SubuMadeUp-014	01/01/2005	0645411340	Office	0	2,456
2	SubuMadeUp-015	01/01/2005	0645411240	Commercial Retail	0	1,300
Residential						
1	SubuMadeUp-016	01/01/2005	0645411060	Multi-Family	4	0
				Sub Total	4	3,756

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
Cool County (Rural)						
Non-Residential						
1	COM2003-00319	06/08/2004	2551890000	Institutional	0	6,000
Residential						
1	MOH2003-00170	05/19/2004	2775360000	Mobile Home	1	0
2	MOH2004-00032	04/21/2004	2173870050	Mobile Home	1	0
3	MOH2004-00034	04/15/2004	1158200000	Mobile Home	1	0
4	MOH2004-00040	04/28/2004	1734640000	Mobile Home	1	0
5	MOH2004-00041	06/07/2004	2569330000	Mobile Home	1	0
6	MOH2004-00056	05/14/2004	1976530000	Mobile Home	1	0
7	MOH2004-00060	06/21/2004	2615810000	Mobile Home	1	0
8	MOH2004-00071	06/08/2004	1683910000	Mobile Home	1	0
9	SFR2003-01425	04/08/2004	1928160000	Single Family	1	0
10	SFR2003-01711	04/08/2004	2213290000	Single Family	1	0
11	SFR2003-01852	04/29/2004	2066851520	Single Family	1	0
12	SFR2003-01987	05/04/2004	2239630000	Single Family	1	0
13	SFR2003-02101	04/01/2004	2567620000	Single Family	1	0
14	SFR2004-00011	05/18/2004	1369070000	Single Family	1	0
15	SFR2004-00080	04/14/2004	2056450000	Single Family	1	0
16	SFR2004-00096	04/23/2004	2667670000	Single Family	1	0
17	SFR2004-00107	04/28/2004	1420210000	Single Family	1	0
18	SFR2004-00112	04/01/2004	1339000000	Single Family	1	0
19	SFR2004-00141	04/15/2004	1368560000	Single Family	1	0
20	SFR2004-00152	05/19/2004	1714870000	Single Family	1	0
21	SFR2004-00155	04/14/2004	2057070000	Additions	1	0
22	SFR2004-00156	05/17/2004	1731570000	Additions	1	0
23	SFR2004-00162	04/06/2004	2036240000	Duplex	2	0
24	SFR2004-00188	04/01/2004	2146810100	Duplex	2	0
25	SFR2004-00202	04/23/2004	1172700000	Single Family	1	0
26	SFR2004-00203	04/13/2004	2036300000	Single Family	1	0
27	SFR2004-00215	06/18/2004	2240830000	Single Family	1	0
28	SFR2004-00217	04/19/2004	2151390060	Single Family	1	0
29	SFR2004-00218	05/10/2004	2544180000	Single Family	1	0
30	SFR2004-00228	04/02/2004	2212570050	Single Family	1	0
31	SFR2004-00235	04/22/2004	1747830000	Single Family	1	0
32	SFR2004-00236	04/16/2004	2353910000	Single Family	1	0
33	SFR2004-00237	05/11/2004	2646390000	Single Family	1	0
34	SFR2004-00246	06/04/2004	2726500000	Single Family	1	0
35	SFR2004-00247	05/03/2004	2043310000	Single Family	1	0
36	SFR2004-00248	04/05/2004	2112730000	Single Family	1	0
37	SFR2004-00253	04/02/2004	1783790000	Single Family	1	0
38	SFR2004-00260	04/26/2004	1410650000	Single Family	1	0
39	SFR2004-00274	04/22/2004	1731560260	Single Family	1	0
40	SFR2004-00275	04/01/2004	2023070220	Single Family	1	0
41	SFR2004-00277	04/08/2004	2276960000	Single Family	1	0
42	SFR2004-00293	04/20/2004	1299050000	Single Family	1	0
43	SFR2004-00294	04/28/2004	1393860000	Single Family	1	0
44	SFR2004-00304	04/06/2004	1901350020	Single Family	1	0
45	SFR2004-00306	05/17/2004	1406900000	Single Family	1	0
46	SFR2004-00310	05/17/2004	1794390000	Single Family	1	0
47	SFR2004-00313	06/10/2004	1330300000	Single Family	1	0
48	SFR2004-00314	04/15/2004	1939460140	Single Family	1	0
49	SFR2004-00320	05/10/2004	1406940000	Single Family	1	0
50	SFR2004-00321	04/12/2004	1939460160	Single Family	1	0
51	SFR2004-00340	06/08/2004	2365880000	Single Family	1	0
52	SFR2004-00341	04/27/2004	1952320000	Single Family	1	0
53	SFR2004-00342	04/30/2004	2016130140	Single Family	1	0
54	SFR2004-00351	05/13/2004	2234840000	Single Family	1	0
55	SFR2004-00355	05/04/2004	2326850000	Single Family	1	0
56	SFR2004-00378	04/26/2004	2531490000	Single Family	1	0

	Permit No	Date	Parcel No	Permit Type	#Units	Non-Res SqFt
57	SFR2004-00379	06/22/2004	2522440000	Single Family	1	0
58	SFR2004-00381	05/27/2004	2339260000	Single Family	1	0
59	SFR2004-00382	05/19/2004	2339070000	Single Family	1	0
60	SFR2004-00383	04/16/2004	1939460080	Single Family	1	0
61	SFR2004-00394	04/22/2004	2266980050	Single Family	1	0
62	SFR2004-00395	04/28/2004	1716960320	Single Family	1	0
63	SFR2004-00400	05/07/2004	1300770050	Single Family	1	0
64	SFR2004-00401	04/28/2004	2232440000	Single Family	1	0
65	SFR2004-00417	05/10/2004	1417960000	Single Family	1	0
66	SFR2004-00422	06/29/2004	1796960000	Single Family	1	0
67	SFR2004-00431	05/12/2004	2051620000	Single Family	1	0
68	SFR2004-00449	06/09/2004	2077710000	Single Family	1	0
69	SFR2004-00464	05/27/2004	2777430000	Single Family	1	0
70	SFR2004-00466	05/26/2004	1808300050	Single Family	1	0
71	SFR2004-00475	04/22/2004	2245810000	Single Family	1	0
72	SFR2004-00476	05/13/2004	1901460040	Single Family	1	0
73	SFR2004-00477	04/30/2004	2368410160	Single Family	1	0
74	SFR2004-00479	05/21/2004	2022940000	Single Family	1	0
75	SFR2004-00480	06/14/2004	2119050000	Single Family	1	0
76	SFR2004-00499	06/21/2004	2777500000	Single Family	1	0
77	SFR2004-00510	05/06/2004	2613200100	Single Family	1	0
78	SFR2004-00512	06/01/2004	1369320000	Single Family	1	0
79	SFR2004-00518	06/04/2004	1300770000	Single Family	1	0
80	SFR2004-00535	04/30/2004	1961650000	Single Family	1	0
81	SFR2004-00561	04/30/2004	2594070000	Single Family	1	0
82	SFR2004-00564	06/24/2004	1959460000	Single Family	1	0
83	SFR2004-00565	05/04/2004	2665880000	Single Family	1	0
84	SFR2004-00579	06/04/2004	2524400000	Single Family	1	0
85	SFR2004-00588	05/25/2004	2096870000	Single Family	1	0
86	SFR2004-00617	06/03/2004	2239620000	Single Family	1	0
87	SFR2004-00622	06/09/2004	2245390000	Single Family	1	0
88	SFR2004-00624	06/14/2004	2035610020	Single Family	1	0
89	SFR2004-00626	06/24/2004	2594010000	Single Family	1	0
90	SFR2004-00632	06/29/2004	2018910000	Single Family	1	0
91	SFR2004-00642	06/03/2004	1417680000	Single Family	1	0
92	SFR2004-00664	05/28/2004	1309740100	Single Family	1	0
93	SFR2004-00666	05/25/2004	1939460020	Single Family	1	0
94	SFR2004-00667	06/18/2004	1976640000	Single Family	1	0
95	SFR2004-00668	05/14/2004	2247110000	Single Family	1	0
96	SFR2004-00669	06/11/2004	2743890000	Single Family	1	0
97	SFR2004-00678	05/21/2004	2481670000	Single Family	1	0
98	SFR2004-00692	04/28/2004	2491340100	Single Family	1	0
99	SFR2004-00708	06/16/2004	2225870000	Single Family	1	0
100	SFR2004-00709	06/15/2004	2666200000	Single Family	1	0
101	SFR2004-00712	06/28/2004	1704320000	Single Family	1	0
102	SFR2004-00724	06/25/2004	2661540000	Single Family	1	0
103	SFR2004-00727	06/22/2004	2265680000	Single Family	1	0
104	SFR2004-00728	06/16/2004	1399280000	Single Family	1	0
105	SFR2004-00729	06/14/2004	2066800000	Single Family	1	0
106	SFR2004-00749	06/21/2004	2334930000	Single Family	1	0
107	SFR2004-00750	06/03/2004	2666370000	Single Family	1	0
108	SFR2004-00751	06/03/2004	2770590000	Single Family	1	0
109	SFR2004-00757	06/24/2004	1847580000	Single Family	1	0
110	SFR2004-00759	05/24/2004	1788900000	Single Family	1	0
111	SFR2004-00760	06/11/2004	1156440600	Single Family	1	0
112	SFR2004-00769	06/04/2004	1800820000	Single Family	1	0
113	SFR2004-00773	06/18/2004	1963110000	Single Family	1	0
114	SFR2004-00785	06/29/2004	2056890000	Single Family	1	0
115	SFR2004-00789	06/30/2004	1398100100	Single Family	1	0
116	SFR2004-00790	06/21/2004	1172800000	Single Family	1	0

						Non-Res
	Permit No	**Date**	**Parcel No**	**Permit Type**	**#Units**	**SqFt**
117	SFR2004-00816	05/24/2004	2670660000	Single Family	1	0
118	SFR2004-00825	06/17/2004	1788440000	Single Family	1	0
119	SFR2004-00832	05/25/2004	2055880150	Single Family	1	0
120	SFR2004-00837	06/15/2004	1996370320	Single Family	1	0
121	SFR2004-00849	06/21/2004	2090630000	Single Family	1	0
122	SFR2004-00946	06/22/2004	2232640000	Single Family	1	0
123	SFR2004-00971	06/11/2004	2567280000	Single Family	1	0
124	SFR2004-00978	06/14/2004	2023070420	Single Family	1	0
125	SFR2004-01030	06/14/2004	1422600000	Single Family	1	0
126	SFR2004-01031	06/24/2004	2036250000	Single Family	1	0
127	SFR2004-01056	06/29/2004	1740600000	Single Family	1	0
128	SFR2004-01075	06/23/2004	2133400000	Single Family	1	0
129	SFR2004-01163	06/25/2004	1697370050	Single Family	1	0
130	SFR2004-01212	06/28/2004	1926640000	Single Family	1	0
				Sub Total	132	6,000
				Total All Jurisdictions:	**1,337**	**857,771**

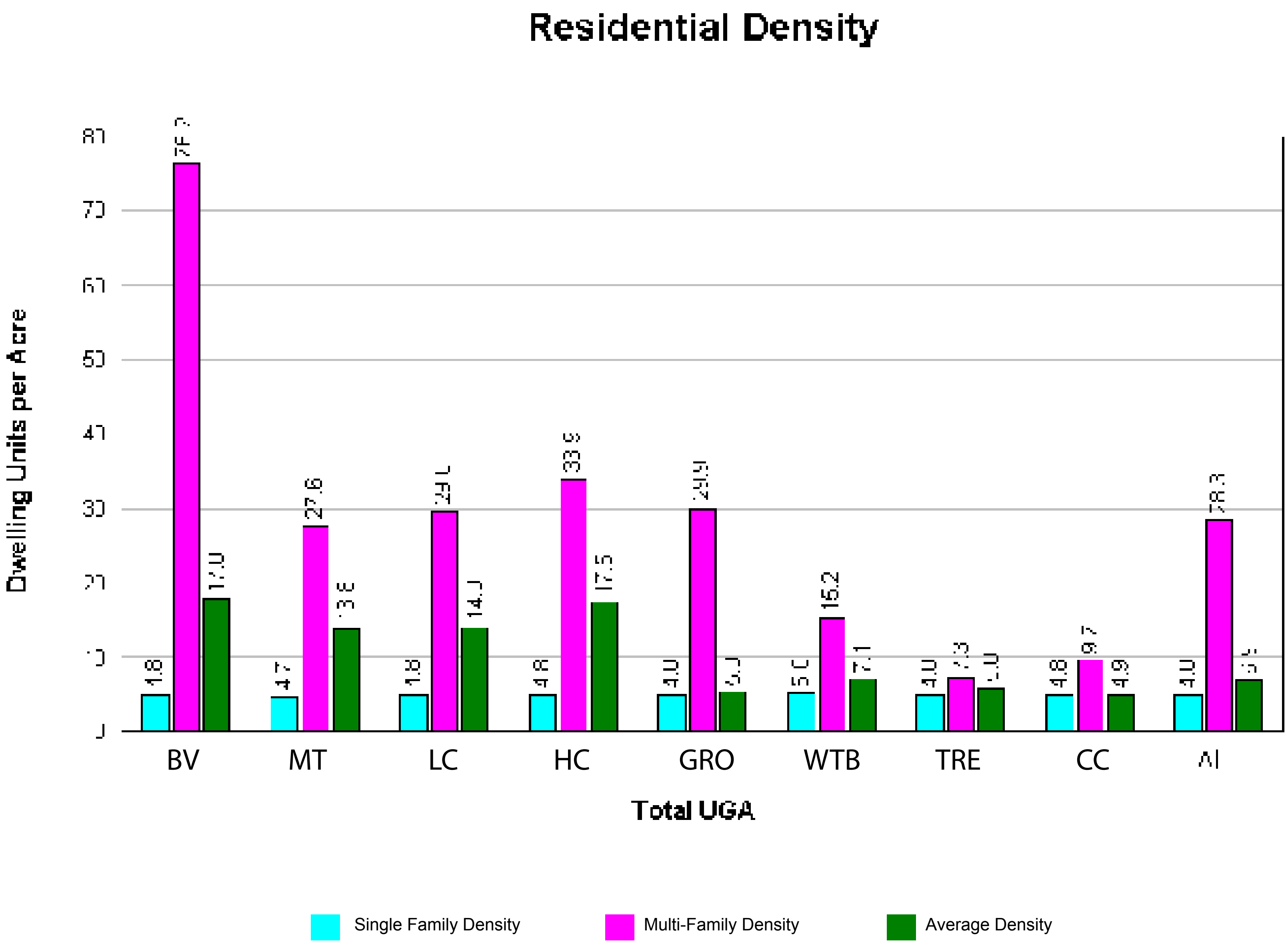
APPENDIX B

SAMPLE RESIDENTIAL SUMMARY REPORT

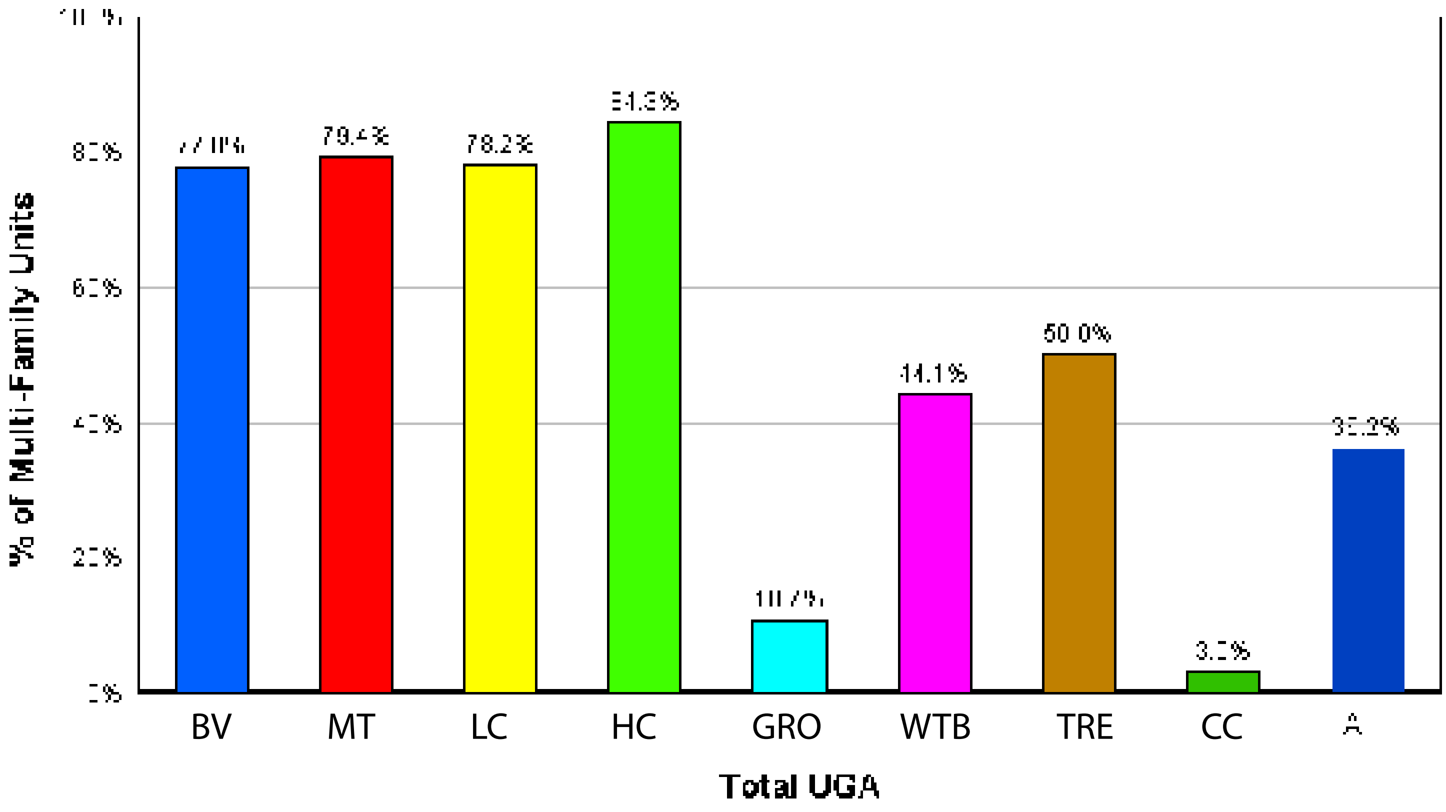
Residential Summary

Reporing Period: 3/1/2004 - 5/31/2005

Jurisdiction	Single Family					Multi-Family						
	#Units			*Acres*	*Net Density*	#Units			*Acres*	*Net Density*	*Average Density*	*% Multi-Family*
	SFR	*ADU*	*MOH*			*MFR*	*DUP*	*MHP*				
Berryville	20	0	0	4.13	4.8	30	0	0	0.62	48.4	10.5	60.0%
Berryville UGA	0	0	0	0.00	0.0	0	0	40	0.30	134.0	134.0	100.0%
Sub Total	20	0	0	4.13	4.8	30	0	40	0.92	76.2	17.8	77.8%
Milltown	31	0	8	8.26	4.7	36	42	76	5.58	27.6	13.9	79.8%
Milltown UGA	1	0	0	0.21	4.8	0	0	0	0.00	0.0	4.8	0.0%
Sub Total	32	0	8	8.47	4.7	36	42	76	5.58	27.6	13.8	79.4%
La Casino	18	0	1	3.93	4.8	54	4	0	1.65	35.1	13.8	75.3%
La Casino UGA	0	0	0	0.00	0.0	8	2	0	0.64	15.6	15.6	100.0%
Sub Total	18	0	1	3.93	4.8	62	6	0	2.30	29.6	14.0	78.2%
Heron City	11	0	2	2.69	4.8	42	2	20	1.56	41.0	18.1	83.1%
Heron City UGA	0	0	0	0.00	0.0	6	0	0	0.51	11.9	11.9	100.0%
Sub Total	11	0	2	2.69	4.8	48	2	20	2.07	33.9	17.5	84.3%
Gromania	129	0	1	26.86	4.8	18	4	0	0.83	26.6	5.5	14.5%
Gromania UGA	438	0	2	90.91	4.8	46	0	0	1.45	31.8	5.3	9.5%
Sub Total	567	0	3	117.77	4.8	64	4	0	2.27	29.9	5.3	10.7%
Waterburg	44	5	5	10.74	5.0	30	10	5	2.96	15.2	7.2	45.5%
Waterburg UGA	3	0	0	0.62	4.8	0	0	0	0.00	0.0	4.8	0.0%
Sub Total	47	5	5	11.36	5.0	30	10	5	2.96	15.2	7.1	44.1%
Treehaven	4	0	0	0.83	4.8	0	0	0	0.00	0.0	4.8	0.0%
Treehaven UGA	0	0	0	0.00	0.0	4	0	0	0.55	7.3	7.3	100.0%
Sub Total	4	0	0	0.83	4.8	4	0	0	0.55	7.3	5.8	50.0%
Cool County (Rural)	118	2	8	26.45	4.8	0	4	0	0.41	9.7	4.9	3.0%
Sub Total (Cities)	257	5	17	57.44	4.9	210	62	101	13.20	28.3	9.2	42.5%
Sub Total (UGAs)	442	0	2	91.74	4.8	64	2	40	3.44	30.8	5.8	17.7%
Sub Total (Urban)	699	5	19	149.17	4.8	274	64	141	16.64	28.8	7.2	31.5%
Grand Total (Urban & Rural)	**817**	**7**	**27**	**175.62**	**4.8**	**274**	**68**	**141**	**17.06**	**28.3**	**6.9**	**36.2%**


Residential Density
Dwelling Units per Acre
80
70
60
50
40
30
20
10
0
4.8
76.2
17.0
4.7
27.6
13.8
4.8
29.0
14.0
4.8
33.6
17.5
4.0
29.9
5.0
5.0
15.2
7.1
4.0
7.3
6.0
4.8
9.7
4.9
4.0
28.3
7.0
BV
MT
LC
HC
GRO
WTB
TRE
CC
AI
Total UGA
Single Family Density
Multi-Family Density
Average Density

MF Units as a Percentage of Total Units



APPENDIX C

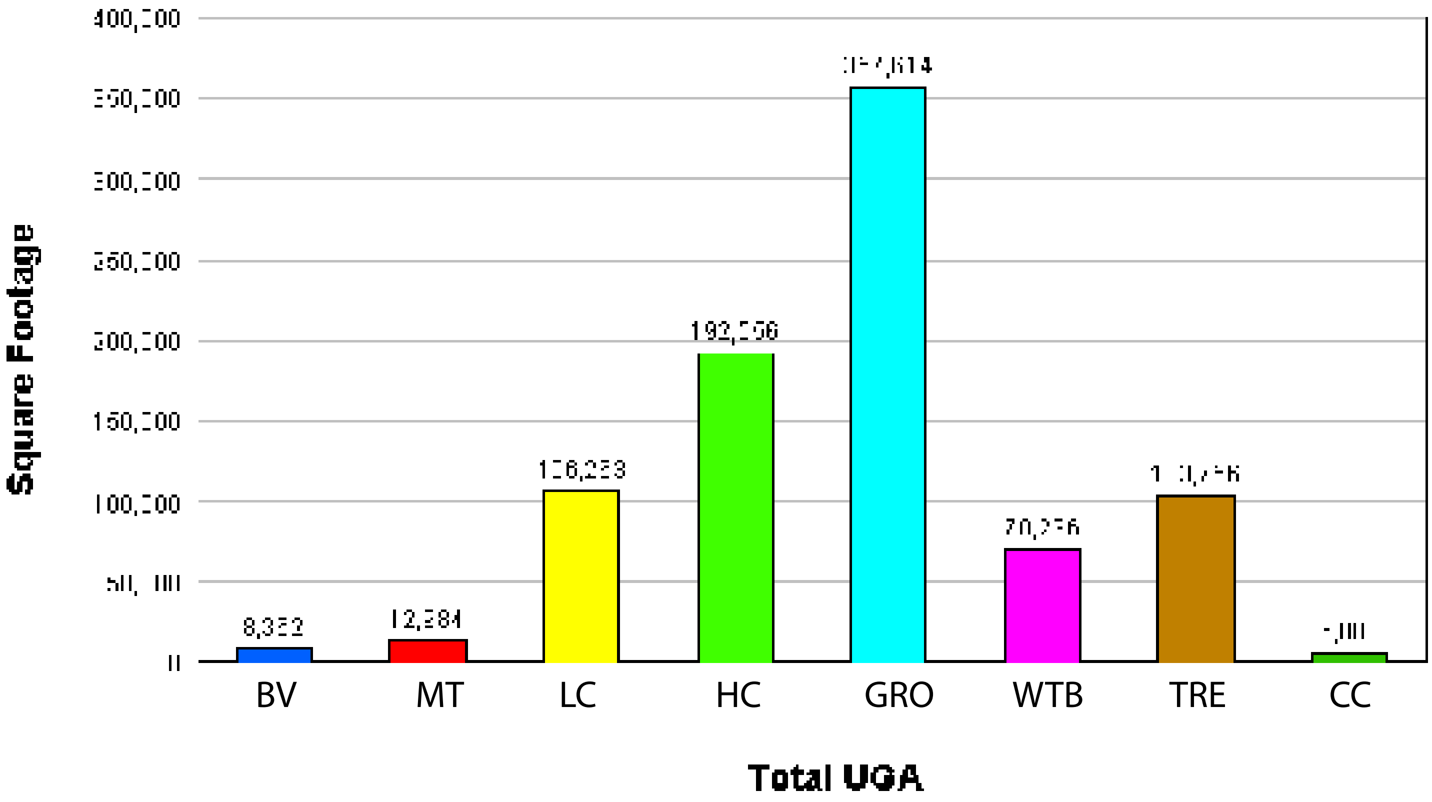
SAMPLE NON-RESIDENTIAL SUMMARY REPORT

Non-Residential Summary

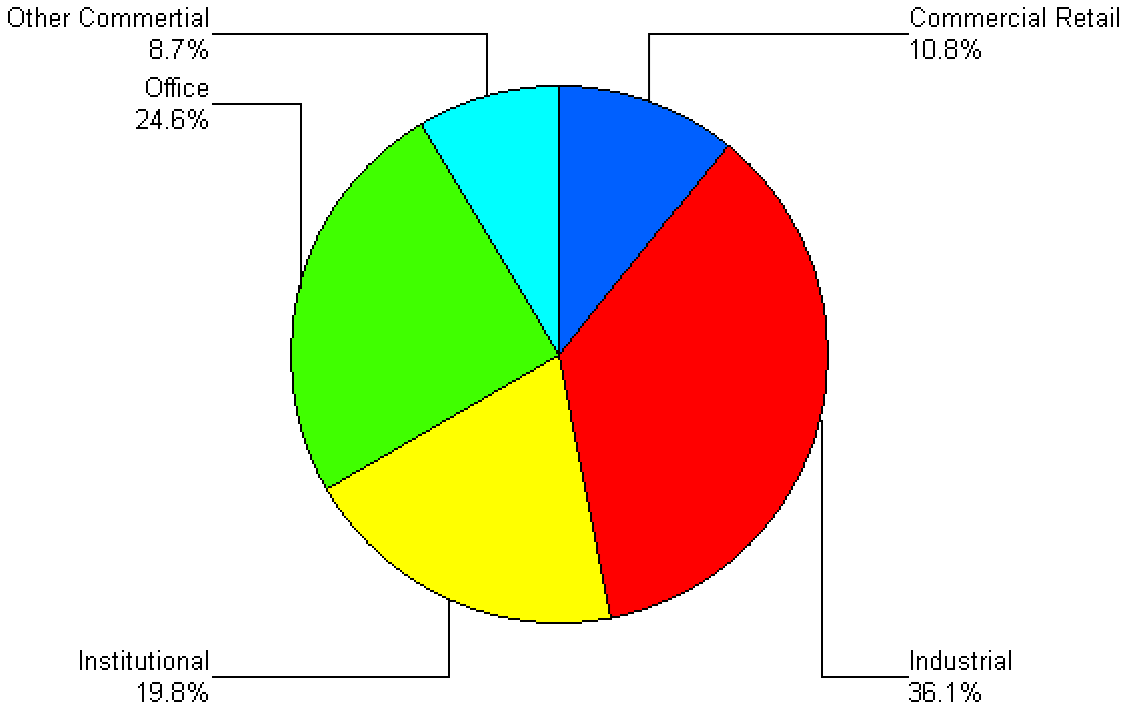
Reporing Period: 3/1/2004 - 5/31/2005

Jurisdiction	Retail		Office		Institutional		Industrial		Others		Total				
	No	*SqFt*	*No*	*SqFt*	*No*	*SqFt*	*No*	*SqFt*	*No*	*SqFt*	*No*	*SqFt*	*Site ac.*	*CA ac.*	*FAR*
Berryville	1	2,561	0	0	0	0	0	0	0	0	1	2,561	0.21	0.02	0.32
Berryville UGA	0	0	0	0	0	0	0	0	2	5,801	2	5,801	0.73	0.04	0.19
Sub Total	1	2,561	0	0	0	0	0	0	2	5,801	3	8,362	0.94	0.06	0.22
Milltown	1	1,240	2	3,195	0	0	1	672	3	2,322	7	7,429	1.45	0.16	0.13
Milltown UGA	1	4,321	0	0	0	0	1	1,234	0	0	2	5,555	0.85	0.16	0.19
Sub Total	2	5,561	2	3,195	0	0	2	1,906	3	2,322	9	12,984	2.30	0.32	0.15
La Casino	5	38,270	5	15,923	0	0	5	24,182	2	8,259	17	86,634	3.51	0.39	0.64
La Casino UGA	1	7,654	0	0	1	6,543	1	5,432	0	0	3	19,629	1.19	0.05	0.39
Sub Total	6	45,924	5	15,923	1	6,543	6	29,614	2	8,259	20	106,263	4.71	0.44	0.57
Heron City	2	5,346	3	7,660	2	70,328	3	56,376	5	34,205	15	173,915	3.28	0.24	1.31
Heron City UGA	1	9,876	1	8,765	0	0	0	0	0	0	2	18,641	0.94	0.13	0.53
Sub Total	3	15,222	4	16,425	2	70,328	3	56,376	5	34,205	17	192,556	4.22	0.37	1.15
Gromania	1	1,357	3	10,547	1	9,753	0	0	0	0	5	21,657	1.03	0.11	0.54
Gromania UGA	3	8,888	4	136,333	4	77,872	6	97,864	1	15,000	18	335,957	3.72	0.22	2.21
Sub Total	4	10,245	7	146,880	5	87,625	6	97,864	1	15,000	23	357,614	4.75	0.34	1.86
Waterburg	5	7,256	3	7,900	2	24,600	1	20,080	3	9,200	14	69,036	2.89	0.32	0.62
Waterburg UGA	1	1,200	0	0	0	0	0	0	0	0	1	1,200	0.24	0.05	0.15
Sub Total	6	8,456	3	7,900	2	24,600	1	20,080	3	9,200	15	70,236	3.13	0.38	0.58
Treeville	0	0	0	0	0	0	1	100,000	0	0	1	100,000	0.21	0.02	12.50
Treeville UGA	1	1,300	1	2,456	0	0	0	0	0	0	2	3,756	0.78	0.24	0.16
Sub Total	1	1,300	1	2,456	0	0	1	100,000	0	0	3	103,756	0.99	0.26	3.27
Cool County (Rural)	0	0	0	0	1	6,000	0	0	0	0	1	6,000	0.21	0.02	0.75
Sub Total (Cities)	15	56,030	16	45,225	5	104,681	11	201,310	13	53,986	60	461,232	12.58	1.27	0.94
Sub Total (UGAs)	8	33,239	6	147,554	5	84,415	8	104,530	3	20,801	30	390,539	8.46	0.90	1.19
Sub Total (Urban)	23	89,269	22	192,779	10	189,096	19	305,840	16	74,787	90	851,771	21.04	2.17	1.04
Grand Total (Urban & Rural)	**23**	**89,269**	**22**	**192,779**	**11**	**195,096**	**19**	**305,840**	**16**	**74,787**	**91**	**857,771**	**21.25**	**2.19**	**1.03**

Non-Residential Square Footage by Jurisdiction



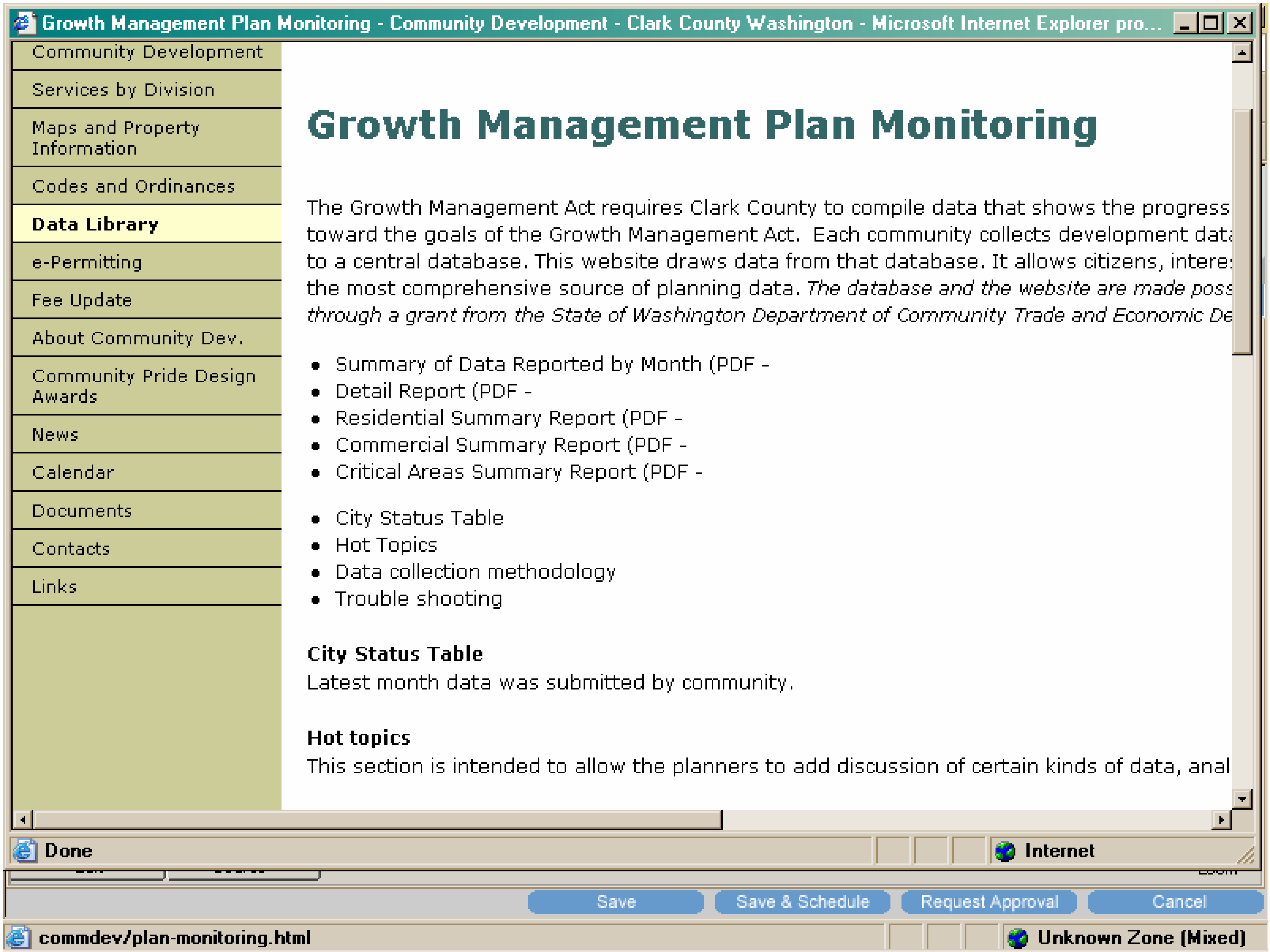
Non-Residential Square Footage by Type



APPENDIX D

SAMPLE WEB PAGE

Appendix D
Sample Web Page



Growth Management Plan Monitoring

The Growth Management Act requires Clark County to compile data that shows the progress toward the goals of the Growth Management Act. Each community collects development dat to a central database. This website draws data from that database. It allows citizens, intere the most comprehensive source of planning data. *The database and the website are made poss through a grant from the State of Washington Department of Community Trade and Economic De*

- Summary of Data Reported by Month (PDF -
- Detail Report (PDF -
- Residential Summary Report (PDF -
- Commercial Summary Report (PDF -
- Critical Areas Summary Report (PDF -

- City Status Table
- Hot Topics
- Data collection methodology
- Trouble shooting

City Status Table
Latest month data was submitted by community.

Hot topics
This section is intended to allow the planners to add discussion of certain kinds of data, anal

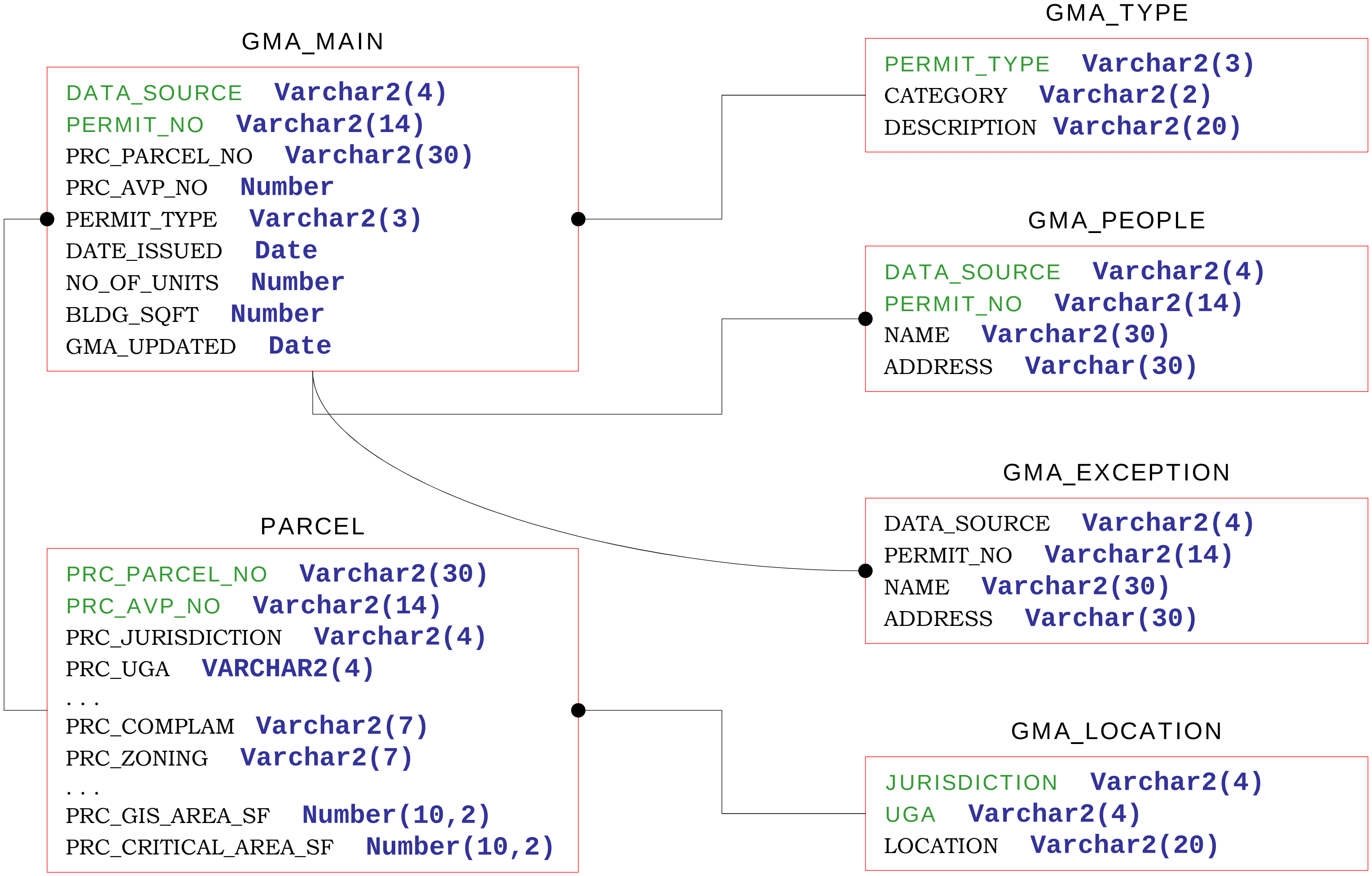
APPENDIX E

PROJECT PROGRAMMING CODE

Clark County Growth Management

Data Repository

E-R Diagram



GMA_MAIN

DATA_SOURCE	Varchar2(4)
PERMIT_NO	Varchar2(14)
PRC_PARCEL_NO	Varchar2(30)
PRC_AVP_NO	Number
PERMIT_TYPE	Varchar2(3)
DATE_ISSUED	Date
NO_OF_UNITS	Number
BLDG_SQFT	Number
GMA_UPDATED	Date

GMA_TYPE

PERMIT_TYPE	Varchar2(3)
CATEGORY	Varchar2(2)
DESCRIPTION	Varchar2(20)

GMA_PEOPLE

DATA_SOURCE	Varchar2(4)
PERMIT_NO	Varchar2(14)
NAME	Varchar2(30)
ADDRESS	Varchar(30)

GMA_EXCEPTION

DATA_SOURCE	Varchar2(4)
PERMIT_NO	Varchar2(14)
NAME	Varchar2(30)
ADDRESS	Varchar(30)

PARCEL

PRC_PARCEL_NO	Varchar2(30)
PRC_AVP_NO	Varchar2(14)
PRC_JURISDICTION	Varchar2(4)
PRC_UGA	VARCHAR2(4)
. . .	
PRC_COMPLAM	Varchar2(7)
PRC_ZONING	Varchar2(7)
. . .	
PRC_GIS_AREA_SF	Number(10,2)
PRC_CRITICAL_AREA_SF	Number(10,2)

GMA_LOCATION

JURISDICTION	Varchar2(4)
UGA	Varchar2(4)
LOCATION	Varchar2(20)

```
CREATE OR REPLACE Procedure GMA_RES_Summary (
  crCursor IN OUT CRPT.refCursor,
  Start_Date IN Date,
  End_Date In Date
)

/*
 +-----------------------------------------------------------------------------
+
 |
|
 |                      Clark County Growth Management
|
 |
|
 |
|
 |                  Procedure Name  :  GMA_RES_Summary
|
 |                         Version  :  1.0
|
 |                    Release Date  :  06/14/2005
|
 |                        Database  :  Oracle 8.1.7 (Release 3)
|
 |                 Operating System :  HP-UX
|
 |                          Author  :  Subu Swayam
|
 |                      Department  :  Information Services,
|
 |                                     Clark County, WA
|
 |
|
 |  Purpose
|
 |   To be able to obtain Residential summary data and be able to produce
|
 |     Crystal Reports with complex Graphics
|
 |
|
 +-----------------------------------------------------------------------------
+
*/

Is

  no_SF_Units Number := 0;
  no_SF_Acres Number := 0;
  no_MF_Units Number := 0;
  no_MF_Acres Number := 0;

  SF_Density_BG Number := 0;
  SF_Density_CAM Number := 0;
  SF_Density_LAC Number := 0;
  SF_Density_RDG Number := 0;
  SF_Density_VAN Number := 0;
  SF_Density_WAS Number := 0;
  SF_Density_YAC Number := 0;
  SF_Density_CC Number := 0;
  SF_Density_ALL Number := 0;

  MF_Density_BG Number := 0;
  MF_Density_CAM Number := 0;
  MF_Density_LAC Number := 0;
  MF_Density_RDG Number := 0;
  MF_Density_VAN Number := 0;
  MF_Density_WAS Number := 0;
  MF_Density_YAC Number := 0;
  MF_Density_CC Number := 0;
  MF_Density_ALL Number := 0;

  AV_Density_BG Number := 0;
  AV_Density_CAM Number := 0;
  AV_Density_LAC Number := 0;
  AV_Density_RDG Number := 0;
  AV_Density_VAN Number := 0;
  AV_Density_WAS Number := 0;
  AV_Density_YAC Number := 0;
  AV_Density_CC Number := 0;
  AV_Density_ALL Number := 0;

  MF_Percent_BG Number := 0;
  MF_Percent_CAM Number := 0;
  MF_Percent_LAC Number := 0;
  MF_Percent_RDG Number := 0;
  MF_Percent_VAN Number := 0;
  MF_Percent_WAS Number := 0;
  MF_Percent_YAC Number := 0;
  MF_Percent_CC Number := 0;
  MF_Percent_ALL Number := 0;

Begin

/*
  --------------
  Summary for BG
  --------------
*/
  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_SF_Units,  no_SF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'BG'
  And T.CATEGORY = 'SF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_MF_Units,  no_MF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'BG'
  And T.CATEGORY = 'MF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  If( no_SF_Acres <> 0 ) Then
    SF_Density_BG := no_SF_Units / no_SF_Acres;
  End If;
  If( no_MF_Acres <> 0 ) Then
    MF_Density_BG := no_MF_Units / no_MF_Acres;
  End If;

  -- Calculate Average Density
  If ( no_SF_Acres + no_MF_Acres ) <> 0 Then
    AV_Density_BG := ( no_SF_Units + no_MF_Units ) / ( no_SF_Acres +
no_MF_Acres );
  End If;

  If ( no_SF_Units + no_MF_Units ) <> 0 Then
    MF_Percent_BG := no_MF_Units / ( no_SF_Units + no_MF_Units );
  End If;

/*
  ---------------
  Summary for CAM
  ---------------
*/
  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_SF_Units,  no_SF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'CAM'
  And T.CATEGORY = 'SF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_MF_Units,  no_MF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'CAM'
  And T.CATEGORY = 'MF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  If( no_SF_Acres <> 0 ) Then
    SF_Density_CAM := no_SF_Units / no_SF_Acres;
  End If;

  If( no_MF_Acres <> 0 ) Then
    MF_Density_CAM := no_MF_Units / no_MF_Acres;
  End If;

  If ( no_SF_Acres + no_MF_Acres ) <> 0 Then
    AV_Density_CAM := ( no_SF_Units + no_MF_Units ) / ( no_SF_Acres +
no_MF_Acres );
  End If;

  If ( no_SF_Units + no_MF_Units ) <> 0 Then
    MF_Percent_CAM := no_MF_Units / ( no_SF_Units + no_MF_Units );
  End If;

/*
  ---------------
  Summary for LAC
  ---------------
*/
  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_SF_Units,  no_SF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'LAC'
  And T.CATEGORY = 'SF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_MF_Units,  no_MF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'LAC'
  And T.CATEGORY = 'MF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  If( no_SF_Acres <> 0 ) Then
    SF_Density_LAC := no_SF_Units / no_SF_Acres;
  End If;

  If( no_MF_Acres <> 0 ) Then
    MF_Density_LAC := no_MF_Units / no_MF_Acres;
  End If;

  If ( no_SF_Acres + no_MF_Acres ) <> 0 Then
    AV_Density_LAC := ( no_SF_Units + no_MF_Units ) / ( no_SF_Acres +
no_MF_Acres );
  End If;

  If ( no_SF_Units + no_MF_Units ) <> 0 Then
    MF_Percent_LAC := no_MF_Units / ( no_SF_Units + no_MF_Units );
  End If;

/*
  ---------------
  Summary for RDG
  ---------------
*/
  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_SF_Units,  no_SF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'RDG'
  And T.CATEGORY = 'SF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_MF_Units,  no_MF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'RDG'
  And T.CATEGORY = 'MF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  If( no_SF_Acres <> 0 ) Then
    SF_Density_RDG := no_SF_Units / no_SF_Acres;
  End If;
  If( no_MF_Acres <> 0 ) Then
    MF_Density_RDG := no_MF_Units / no_MF_Acres;
  End If;

  If ( no_SF_Acres + no_MF_Acres ) <> 0 Then
    AV_Density_RDG := ( no_SF_Units + no_MF_Units ) / ( no_SF_Acres +
no_MF_Acres );
  End If;

  If ( no_SF_Units + no_MF_Units ) <> 0 Then
    MF_Percent_RDG := no_MF_Units / ( no_SF_Units + no_MF_Units );
  End If;

/*
  ---------------
  Summary for VAN
  ---------------
*/
  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_SF_Units,  no_SF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'VAN'
  And T.CATEGORY = 'SF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_MF_Units,  no_MF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'VAN'
  And T.CATEGORY = 'MF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  If( no_SF_Acres <> 0 ) Then
    SF_Density_VAN := no_SF_Units / no_SF_Acres;
  End If;

  If( no_MF_Acres <> 0 ) Then
    MF_Density_VAN := no_MF_Units / no_MF_Acres;
  End If;

  If ( no_SF_Acres + no_MF_Acres ) <> 0 Then
    AV_Density_VAN := ( no_SF_Units + no_MF_Units ) / ( no_SF_Acres +
no_MF_Acres );
  End If;

  If ( no_SF_Units + no_MF_Units ) <> 0 Then
    MF_Percent_VAN := no_MF_Units / ( no_SF_Units + no_MF_Units );
  End If;


/*
  ---------------
  Summary for WAS
  ---------------
*/
  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_SF_Units,  no_SF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'WAS'
  And T.CATEGORY = 'SF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_MF_Units,  no_MF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'WAS'
  And T.CATEGORY = 'MF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  If( no_SF_Acres <> 0 ) Then
    SF_Density_WAS := no_SF_Units / no_SF_Acres;
  End If;

  If( no_MF_Acres <> 0 ) Then
    MF_Density_WAS := no_MF_Units / no_MF_Acres;
  End If;

  If ( no_SF_Acres + no_MF_Acres ) <> 0 Then
    AV_Density_WAS := ( no_SF_Units + no_MF_Units ) / ( no_SF_Acres +
no_MF_Acres );
  End If;

  If ( no_SF_Units + no_MF_Units ) <> 0 Then
    MF_Percent_WAS := no_MF_Units / ( no_SF_Units + no_MF_Units );
  End If;


/*
  ---------------
  Summary for YAC
  ---------------
*/
  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_SF_Units,  no_SF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'YAC'
  And T.CATEGORY = 'SF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_MF_Units,  no_MF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'YAC'
  And T.CATEGORY = 'MF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  If( no_SF_Acres <> 0 ) Then
    SF_Density_YAC := no_SF_Units / no_SF_Acres;
  End If;
  If( no_MF_Acres <> 0 ) Then
    MF_Density_YAC := no_MF_Units / no_MF_Acres;
  End If;

  If ( no_SF_Acres + no_MF_Acres ) <> 0 Then
    AV_Density_YAC := ( no_SF_Units + no_MF_Units ) / ( no_SF_Acres +
no_MF_Acres );
  End If;

  If ( no_SF_Units + no_MF_Units ) <> 0 Then
    MF_Percent_YAC := no_MF_Units / ( no_SF_Units + no_MF_Units );
  End If;


/*
  --------------
  Summary for CC
  --------------
*/
  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_SF_Units,  no_SF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'CC'
  And T.CATEGORY = 'SF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_MF_Units,  no_MF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And P.PRC_URB_GRWTH_BNDRY = 'CC'
  And T.CATEGORY = 'MF'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  If( no_SF_Acres <> 0 ) Then
    SF_Density_CC := no_SF_Units / no_SF_Acres;
  End If;

  If( no_MF_Acres <> 0 ) Then
    MF_Density_CC := no_MF_Units / no_MF_Acres;
  End If;

  If ( no_SF_Acres + no_MF_Acres ) <> 0 Then
    AV_Density_CC := ( no_SF_Units + no_MF_Units ) / ( no_SF_Acres +
no_MF_Acres );
  End If;

  If ( no_SF_Units + no_MF_Units ) <> 0 Then
    MF_Percent_CC := no_MF_Units / ( no_SF_Units + no_MF_Units );
  End If;


/*
  ---------------
  Summary for All
  ---------------
*/
  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_SF_Units,  no_SF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And T.CATEGORY = 'SF'
  And P.PRC_URB_GRWTH_BNDRY <> '~'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  Select SUM( M.NO_OF_UNITS ), SUM( P.PRC_GIS_AREA_SF )/43560
  Into no_MF_Units,  no_MF_Acres
  From GMA_MAIN M, PARCEL P, GMA_TYPE T
  Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
  And M.PERMIT_TYPE = T.PERMIT_TYPE
  And T.CATEGORY = 'MF'
  And P.PRC_URB_GRWTH_BNDRY <> '~'
  And M.DATE_ISSUED Between Start_Date And End_Date;

  If( no_SF_Acres <> 0 ) Then
    SF_Density_ALL := no_SF_Units / no_SF_Acres;
  End If;

  If( no_MF_Acres <> 0 ) Then
    MF_Density_ALL := no_MF_Units / no_MF_Acres;
  End If;

  If ( no_SF_Acres + no_MF_Acres ) <> 0 Then
    AV_Density_ALL := ( no_SF_Units + no_MF_Units ) / ( no_SF_Acres +
no_MF_Acres );
  End If;

  If ( no_SF_Units + no_MF_Units ) <> 0 Then
    MF_Percent_ALL := no_MF_Units / ( no_SF_Units + no_MF_Units );
  End If;

/*
  -----------------------
  Provide Values to Cursor
  -----------------------
*/
  Open crCursor For

    Select
    'BG' As UGA,
    SF_Density_BG As SF_DENSITY,
    MF_Density_BG As MF_DENSITY,
    AV_Density_BG As AV_DENSITY,
    MF_Percent_BG As MF_PERCENT
    From Dual

    Union

    Select
    'CAM' As UGA,
    SF_Density_CAM As SF_DENSITY,
    MF_Density_CAM As MF_DENSITY,
    AV_Density_CAM As AV_DENSITY,
    MF_Percent_CAM As MF_PERCENT
    From Dual

    Union
```

```
Select
'LAC' As UGA,
SF_Density_LAC As SF_DENSITY,
MF_Density_LAC As MF_DENSITY,
AV_Density_LAC As AV_DENSITY,
MF_Percent_LAC As MF_PERCENT
From Dual

Union

Select
'RDG' As UGA,
SF_Density_RDG As SF_DENSITY,
MF_Density_RDG As MF_DENSITY,
AV_Density_RDG As AV_DENSITY,
MF_Percent_RDG As MF_PERCENT
From Dual

Union

Select
'VAN' As UGA,
SF_Density_VAN As SF_DENSITY,
MF_Density_VAN As MF_DENSITY,
AV_Density_VAN As AV_DENSITY,
MF_Percent_VAN As MF_PERCENT
From Dual

Union

Select
'WAS' As UGA,
SF_Density_WAS As SF_DENSITY,
MF_Density_WAS As MF_DENSITY,
AV_Density_WAS As AV_DENSITY,
MF_Percent_WAS As MF_PERCENT
From Dual

Union

Select
'YAC' As UGA,
SF_Density_YAC As SF_DENSITY,
MF_Density_YAC As MF_DENSITY,
AV_Density_YAC As AV_DENSITY,
MF_Percent_YAC As MF_PERCENT
From Dual

Union

Select
'CC' As UGA,
SF_Density_CC As SF_DENSITY,
MF_Density_CC As MF_DENSITY,
AV_Density_CC As AV_DENSITY,
MF_Percent_CC As MF_PERCENT
From Dual

Union

Select
```

```
    'ALL' As UGA,
    SF_Density_ALL As SF_DENSITY,
    MF_Density_ALL As MF_DENSITY,
    AV_Density_ALL As AV_DENSITY,
    MF_Percent_ALL As MF_PERCENT
    From Dual;

End GMA_RES_Summary;
/
```

```
CREATE OR REPLACE Package GMA

Is

-- Package Specification
/*
 +-----------------------------------------------------------------------------
+
 |
|
 |                      Clark County Growth Management
|
 |
|
 |
|
 |                    Package Name  :  GMA
|
 |                         Version  :  1.0 Beta
|
 |                    Release Date  :  06/14/2005
|
 |                        Database  :  Oracle 8.1.7 (Release 3)
|
 |                 Operating System :  HP-UX
|
 |                          Author  :  Subu Swayam
|
 |                      Department  :  Information Services,
|
 |                                     Clark County, WA
|
 |
|
 |  Purpose
|
 |   To import following permits issued by various Clark County jurisdictions
|
 |    into a "Data Repository" for use by Long Range Planning.
|
 |
|
 |                 ***********            ***************
|
 |                 Residential            Non-Residential
|
 |                 ***********            ***************
|
 |                 Single Family          Commercial Retail
|
 |                 Addition/Remodel       Office
|
 |                 Mobile Home            Institutional
|
 |                 Multi-Family           Industrial
|
 |                 Duplex                 Others
|
 |                 Mobile Home Park
|
 |
|
 |  --------------------------------------------------------------------------
|
 |  Code Modifications:
|
 |   No     Date     Author   Description
|
 |  ---- ---------- -------- -------------------------------------------------
|
 |
|
 |   01  06/30/2005  SubuS    No changes yet
|
 |
|
 |
|
 +----------------------------------------------------------------------------
+
*/


-- Permit Record
Type rPermit Is Record (
  DataSource PARCEL.PRC_JURISDICTION%Type,
  PermitNo CASEMAIN.CSM_CASENO%Type,
  PermitType CASEMAIN.CASE_Type%Type,
  DateIssued Date,
  ParcelNo PARCEL.PRC_PARCEL_NO%Type,
  NoUnits Number,
  BldgSqFt Number,
  Name GMA_PEOPLE.NAME%Type,
  Address GMA_PEOPLE.ADDRESS%Type
);


-- Field positions in the Input File
Type rPosition Is Record (
  PermitNo_p Number,
  PermitType_p Number,
  DateIssued_p Number,
  ParcelNo_p Number,
  NoUnits_p Number,
  BldgSqFt_p Number,
  Name_p Number,
  Address_p Number
);


/*
-------------
Export_Data()
-------------
  -- INPUT(s) : None
  -- OUTPUT(s) : Comma Separated File with records containing 9 fields
  -- CALLER(s) : This will be called by an UNIX cron job once a month
  -- PURPOSE : To export relevant data from the City and the County
*/
Procedure Export_Data;
```

```
/*
--------------
Extract_SFRs()
--------------
  -- INPUT(s) : None
  -- OUTPUT(s) : Comma Separated Values (CSV) File
  -- CALLER(s) : Export_Data()
  -- PURPOSE : To write SFR records to the export file
*/
Procedure Extract_SFRs;


/*
--------------
Extract_ADUs()
--------------
  -- INPUT(s) : None
  -- OUTPUT(s) : Comma Separated File (CSV)
  -- CALLER(s) : Export_Data()
  -- PURPOSE : To write ADU records to the export file
*/
Procedure Extract_ADUs;


/*
--------------
Extract_MOHs()
--------------
  -- INPUT(s) : None
  -- OUTPUT(s) : Comma Separated File (CSV)
  -- CALLER(s) : Export_Data()
  -- PURPOSE : To write MOH records to the export file
*/
Procedure Extract_MOHs;


/*
--------------
Extract_MFRs()
--------------
  -- INPUT(s) : None
  -- OUTPUT(s) : Comma Separated File (CSV)
  -- CALLER(s) : Export_Data()
  -- PURPOSE : To write MFR records to the export file
*/
Procedure Extract_MFRs;


/*
--------------
Extract_DUPs()
--------------
  -- INPUT(s) : None
  -- OUTPUT(s) : Comma Separated File (CSV)
  -- CALLER(s) : Export_Data()
  -- PURPOSE : To write DUP records to the export file
*/
Procedure Extract_DUPs;
```

```
/*
--------------
Extract_MHPs()
--------------
  -- INPUT(s) : Permit Type - FSR (county) / PSR (city)
  -- OUTPUT(s) : Comma Separated File (CSV)
  -- CALLER(s) : Export_Data()
  -- PURPOSE : To write MHP records to the export file
*/
Procedure Extract_MHPs
(
  PermitType CASEMAIN.CASE_Type%Type
);


/*
--------------
Extract_COMs()
--------------
  -- INPUT(s) : Permit Type, Census Category
  -- OUTPUT(s) : Comma Separated File (CSV)
  -- CALLER(s) : Export_Data()
  -- PURPOSE : To write COM records to the export file
*/
Procedure Extract_COMs
(
  PermitType CASEMAIN.CASE_Type%Type,
  CensusNo CASE_EXTENDED.CSM_CENSUS_CATEGORY%Type
);


/*
-----------
Bldg_SqFt()
-----------
  -- INPUT(s) : Permit No
  -- OUTPUT(s) : Bldg Sqft
  -- CALLER(s) : Extract_COMs
  -- PURPOSE : To calculate non-residential sqft
*/
Function Bldg_SqFt
(
  CaseNo CASEMAIN.CSM_CASENO%Type
)

Return Number;


/*
-----------------
Set_Date_Params()
-----------------
  -- INPUT(s) : None
  -- OUTPUT(s) : Start and End Dates
  -- CALLER(s) : Export_Data
  -- PURPOSE : To calculate date range for data extraction
*/
Procedure Set_Date_Params
```

```
(
  sDate Out DATE,
  eDate Out DATE
);


/*
-------------
Import_Data()
-------------
  -- INPUT(s) : Comma Separated File with records containing 9 fields
  -- OUTPUT(s) : Updates the GMA Data Repository
  -- CALLER(s) : This will be called by an UNIX cron job once a month
  -- PURPOSE : To import relevant data from various Clark County jurisdictions
*/
Procedure Import_Data;

/*
-----------
Load_Data()
-----------
  -- INPUT(s) : Name of the Data Source
  -- OUTPUT(s) : Updates the GMA Data Repository
  -- CALLER(s) : Import_Data()
  -- PURPOSE : To read, translate, process and insert records into the
repository
*/
Procedure Load_Data
(
  DataSource PARCEL.PRC_JURISDICTION%Type
);


/*
-------------
Init_Permit()
-------------
  -- INPUT(s) : Permit Record
  -- OUTPUT(s) : None
  -- CALLER(s) : Load_Data()
  -- PURPOSE : To initialize the record
*/
Procedure Init_Permit
(
  permitR Out rPermit
);


/*
-------------
Read_Permit()
-------------
  -- INPUT(s) : Records from the CSV Files
  -- OUTPUT(s) : Updated Permit Record
  -- CALLER(s) : Load_Data()
  -- PURPOSE : To read, translate and load data into the record structure
*/
Procedure Read_Permit
(
  DataSource PARCEL.PRC_JURISDICTION%Type,
  permitR Out rPermit,
  aRecord Varchar2
);


/*
--------------
Set_Position()
--------------
  -- INPUT(s) : Permit Record
  -- OUTPUT(s) : Updated Permit Record
  -- CALLER(s) : Load_Data()
  -- PURPOSE : To read, translate and load data into the record structure
*/
Procedure Set_Position
(
  DataSource In PARCEL.PRC_JURISDICTION%Type,
  positionR Out rPosition
);


/*
-------------------
Translate_ParcelNo()
-------------------
  -- INPUT(s) : Permit Record, Parcel field position
  -- OUTPUT(s) : Translated Parcel No
  -- CALLER(s) : Read_Permit
  -- PURPOSE : Formats the Parcel No by removing hyphen and makes it 10
charcters long
*/
Function Translate_ParcelNo
(
  aRecord In Varchar2,
  positionR In rPosition
)

Return Varchar2;


/*
----------------------
Translate_PermitType()
----------------------
  -- INPUT(s) : Permit Record, Permit Type field position
  -- OUTPUT(s) : Translated Permit Type
  -- CALLER(s) : Read_Permit
  -- PURPOSE : Returns 3-character Permit Type using the GIS_TRANSLATOR table
*/
Function Translate_PermitType
(
  aRecord In Varchar2,
  positionR In rPosition
)

Return Varchar2;


/*
---------------------
```

```
Parse_Data_By_Comma()
---------------------
  -- INPUT(s) : Records from the CSV Files, field position
  -- OUTPUT(s) : Field data value
  -- CALLER(s) : Read_Permit
  -- PURPOSE : To parse and return the necessary data by field number
*/
Function Parse_Data_By_Comma
(
  aRecord In Varchar2,
  fieldNo In Number
)

Return Varchar2;


/*
---------------
Insert_Permit()
---------------
  -- INPUT(s) : Updated Permit Record
  -- OUTPUT(s) : Updated GMA_MAIN
  -- CALLER(s) : Load_Data()
  -- PURPOSE : To insert the permit record into GMA_MAIN
*/
Function Insert_Permit
(
  permitR In rPermit
)

Return Boolean;


/*
---------------
Insert_People()
---------------
  -- INPUT(s) : Updated Permit Record
  -- OUTPUT(s) : Updated GMA_PEOPLE
  -- CALLER(s) : Load_Data()
  -- PURPOSE : To insert the permit record into GMA_PEOPLE
*/
Function Insert_People
(
  permitR In rPermit
)

Return Boolean;


/*
-------------------
Populate_GIS_Data()
-------------------
  -- INPUT(s) : None
  -- OUTPUT(s) : Updated Parcel table with GIS data
  -- CALLER(s) : This will be called by an UNIX cron job once a month
  -- PURPOSE : Runs MLQ and obtains the XML DOM object in real-time
*/
Procedure Populate_GIS_Data;
```

```
/*
--------------
Get_GIS_Data()
--------------
  -- INPUT(s) : Parcel No
  -- OUTPUT(s) : GIS record with updated values
  -- CALLER(s) : Populate_GIS_Data()
  -- PURPOSE : To parse the XML document and identify the GIS values
*/
Function Get_Gis_Data
(
  ParcelNo In CASEMAIN.PRC_PARCEL_NO%Type,
  gd In Out GIS.rGisData
)

Return Boolean;


End GMA;
/
CREATE OR REPLACE Package Body GMA

Is

  DataSource PARCEL.PRC_JURISDICTION%Type := Null;

  -- Variables for calculating date range for data extraction
  sDate Date := Null;
  eDate Date := Null;

  -- Program Executed Date
  rDate Varchar2( 8 ) := To_Char( Sysdate, 'YYYYMMDD' );

  DirN Varchar2( 15 ) := Null;
  DatFileN Varchar2( 8 ) := Null;
  DatFileP Utl_File.File_Type := Null;
  LogFileN Varchar2( 25 ) := Null;
  LogFileP Utl_File.File_Type := Null;
  ErrFileN Varchar2( 25 ) := Null;
  ErrFileP Utl_File.File_Type := Null;


/*
-------------
Export_Data()
-------------
*/
Procedure Export_Data

Is

  DbInstance Varchar2( 10 ) := Null;

Begin

  -- Identify the Database Instance where the procedure is running
  Select lower( NAME ) Into DbInstance From SYS.v_$database;
  If SUBSTR( DbInstance, 2, 1 ) = 'c' Then
    DataSource := 'CC';
  Else
    DataSource := 'VAN';
  End If;

  DirN := '/v01/' || DbInstance || '/dump';
  DatFileN := DataSource || '.csv';
  DatFileP := Utl_File.Fopen( DirN, DatFileN, 'w' );
  LogFileN := DataSource || '_Export_' || rDate || '.log';
  LogFileP := Utl_File.Fopen( DirN, LogFileN, 'a' );
  ErrFileN := DataSource || '_Export_' || rDate || '.err';

  Utl_File.Put_Line( LogFileP, To_Char( Sysdate, 'HH24:Mi:SS' ) );
  Utl_File.Put_Line( LogFileP, 'Exported the followIng cases:' );
  Utl_File.put_Line( DatFileP, 'DummyHeading' );  -- will be thown away

  Set_Date_Params( sDate, eDate );
  Extract_SFRs;
  Extract_ADUs;
  Extract_MOHs;
  Extract_MFRs;
  Extract_DUPs;
  If DataSource = 'CC' Then
    Extract_MHPs( 'FSR' );
  Else
    Extract_MHPs( 'PSR' );
  End If;
  Extract_COMs( 'RET', '322' );
  Extract_COMs( 'RET', '327' );
  Extract_COMs( 'OFF', '324' );
  Extract_COMs( 'INS', '319' );
  Extract_COMs( 'INS', '323' );
  Extract_COMs( 'INS', '325' );
  Extract_COMs( 'INS', '326' );
  Extract_COMs( 'IND', '320' );
  Extract_COMs( 'COM', '213' );
  Extract_COMs( 'COM', '214' );
  Extract_COMs( 'COM', '318' );
  Extract_COMs( 'COM', '328' );

  Utl_File.Fclose( DatFileP );
  Utl_File.Fclose( LogFileP );
  Utl_File.Fclose( ErrFileP );

End Export_Data;


/*
--------------
Extract_SFRs()
--------------
*/
Procedure Extract_SFRs

Is

  aRecord Varchar2( 255 );
  howMany Number := 0;
  Cursor CaseList Is

    Select C.CSM_CASENO, M.CSM_ISSUED_DATE, M.PRC_PARCEL_NO, E.CSM_NO_OF_UNITS
    From CASE_SFR C, CASEMAIN M, CASE_EXTENDED E
    Where C.CSM_CASENO = M.CSM_CASENO
    And M.CSM_CASENO = E.CSM_CASENO
    And M.CSM_STATUS In ( 'APR', 'FNL' )
    And E.CSM_SCOPE_OF_WORK Not Like 'DUPLEX%'
    And M.CSM_CASENO Not Like '%TEST%'
    And M.CSM_CASENO Not Like '%1900-%'
    And M.CSM_ISSUED_DATE Between sDate And eDate
    Order By M.CSM_CASENO;

Begin

  For EachCase In CaseList
  Loop
    aRecord := DataSource || ',' ||
               EachCase.CSM_CASENO || ',' ||
               'SFR,' ||
               To_Char( EachCase.CSM_ISSUED_DATE, 'MM/DD/YY' ) || ',' ||
               EachCase.PRC_PARCEL_NO || ',' ||
               NVL( EachCase.CSM_NO_OF_UNITS, 1 ) || ',' ||
               '0,,';  -- Non-Residential SqFt, Owner, Address

    Utl_File.Put_Line( DatFileP, aRecord );
    howMany := howMany + 1;
  End Loop;

  Utl_File.Put_Line( LogFileP, '-> SFR : ' || To_Char( howMany ) );

  Exception
    When Others Then
    ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
    Utl_File.Put_Line( ErrFileP, 'Error: Procedure Extract_SFRs' );

End Extract_SFRs;


/*
--------------
Extract_ADUs()
--------------
*/
Procedure Extract_ADUs

Is

  aRecord Varchar2( 255 );
  howMany Number := 0;

  Cursor CaseList Is

    Select C.CSM_CASENO, M.CSM_ISSUED_DATE, M.PRC_PARCEL_NO, E.CSM_NO_OF_UNITS
    From CASE_RES C, CASEMAIN M, CASE_EXTENDED E
    Where C.CSM_CASENO = M.CSM_CASENO
    And M.CSM_CASENO = E.CSM_CASENO
    And M.CSM_STATUS In ( 'APR', 'FNL' )
    And E.CSM_SCOPE_OF_WORK Like 'ADU%'
    And M.CSM_CASENO Not Like '%TEST%'
    And M.CSM_CASENO Not Like '%1900-%'
    And M.CSM_ISSUED_DATE Between sDate And eDate
    Order By M.CSM_CASENO;

Begin

  For EachCase In CaseList
  Loop
    aRecord := DataSource || ',' ||
               EachCase.CSM_CASENO || ',' ||
               'ADU,' ||
               To_Char( EachCase.CSM_ISSUED_DATE, 'MM/DD/YY' ) || ',' ||
               EachCase.PRC_PARCEL_NO || ',' ||
               NVL( EachCase.CSM_NO_OF_UNITS, 1 ) || ',' ||
               '0,,';  -- Non-Residential SqFt, Owner, Address

    Utl_File.Put_Line( DatFileP, aRecord );
    howMany := howMany + 1;
  End Loop;

  Utl_File.Put_Line( LogFileP, '-> ADU : ' || To_Char( howMany ) );

  Exception
    When Others Then
    ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
    Utl_File.Put_Line( ErrFileP, 'Error: Procedure Extract_SFRs' );

End Extract_ADUs;


/*
--------------
Extract_MOHs()
--------------
*/
Procedure Extract_MOHs

Is

  aRecord Varchar2( 255 );
  howMany Number := 0;

  Cursor CaseList Is

    Select H.CSM_CASENO, M.CSM_ISSUED_DATE, M.PRC_PARCEL_NO, E.CSM_NO_OF_UNITS
    From CASE_MOH H, CASEMAIN M, CASE_EXTENDED E
    Where H.CSM_CASENO = M.CSM_CASENO
    And M.CSM_CASENO = E.CSM_CASENO
    And H.MOH_TYPE_PLACEMENT In ( 'PERM', 'MOD' )
    And H.MOH_PARK_ACREAGE = 'N'
    And M.CSM_STATUS In ( 'APR', 'FNL' )
    And M.CSM_CASENO Not Like '%TEST%'
    And M.CSM_CASENO Not Like '%1900-%'
    And M.CSM_ISSUED_DATE Between sDate And eDate
    Order By M.CSM_CASENO;

Begin

  For EachCase In CaseList
  Loop
```

```
    aRecord := DataSource || ',' ||
               EachCase.CSM_CASENO || ',' ||
               'MOH,' ||
               To_Char( EachCase.CSM_ISSUED_DATE, 'MM/DD/YY' ) || ',' ||
               EachCase.PRC_PARCEL_NO || ',' ||
               NVL( EachCase.CSM_NO_OF_UNITS, 1 ) || ',' ||
               '0,,';  -- Non-Residential SqFt, Owner, Address

    Utl_File.Put_Line( DatFileP, aRecord );
    howMany := howMany + 1;
  End Loop;

  Utl_File.Put_Line( LogFileP, '-> MOH : ' || To_Char( howMany ) );

  Exception
    When Others Then
    ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
    Utl_File.Put_Line( ErrFileP, 'Error: Procedure Extract_MOHs' );

End Extract_MOHs;


/*
--------------
Extract_MFRs()
--------------
*/
Procedure Extract_MFRs

Is

  aRecord Varchar2( 255 );
  howMany Number := 0;

  Cursor CaseList Is

    Select F.CSM_CASENO, M.CSM_ISSUED_DATE, M.PRC_PARCEL_NO, E.CSM_NO_OF_UNITS
    From CASE_MFR F, CASEMAIN M, CASE_EXTENDED E
    Where F.CSM_CASENO = M.CSM_CASENO
    And M.CSM_CASENO = E.CSM_CASENO
    And E.CSM_CENSUS_CATEGORY In ( '104', '105' )
    And M.CSM_STATUS In ( 'APR', 'FNL' )
    And M.CSM_CASENO Not Like '%TEST%'
    And M.CSM_CASENO Not Like '%1900-%'
    And M.CSM_ISSUED_DATE Between sDate And eDate
    Order By M.CSM_CASENO;

Begin

  For EachCase In CaseList
  Loop
    aRecord := DataSource || ',' ||
               EachCase.CSM_CASENO || ',' ||
               'MFR,' ||
               To_Char( EachCase.CSM_ISSUED_DATE, 'MM/DD/YY' ) || ',' ||
               EachCase.PRC_PARCEL_NO || ',' ||
               NVL( EachCase.CSM_NO_OF_UNITS, 0 ) || ',' ||
               '0,,';  -- Non-Residential SqFt, Owner, Address

    Utl_File.Put_Line( DatFileP, aRecord );
    howMany := howMany + 1;
  End Loop;

  Utl_File.Put_Line( LogFileP, '-> MFR : ' || To_Char( howMany ) );

  Exception
    When Others Then
    ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
    Utl_File.Put_Line( ErrFileP, 'Error: Procedure Extract_MFRs' );

End Extract_MFRs;


/*
--------------
Extract_DUPs()
--------------
*/
Procedure Extract_DUPs

Is

  aRecord Varchar2( 255 );
  howMany Number := 0;

  Cursor CaseList Is

    Select S.CSM_CASENO, M.CSM_ISSUED_DATE, M.PRC_PARCEL_NO, E.CSM_NO_OF_UNITS
    From CASE_SFR S, CASEMAIN M, CASE_EXTENDED E
    Where S.CSM_CASENO = M.CSM_CASENO
    And M.CSM_CASENO = E.CSM_CASENO
    And M.CSM_STATUS In ( 'APR', 'FNL' )
    And E.CSM_SCOPE_OF_WORK Like 'DUPLEX%'
    And M.CSM_CASENO Not Like '%TEST%'
    And M.CSM_CASENO Not Like '%1900-%'
    And M.CSM_ISSUED_DATE Between sDate And eDate
    Order By M.CSM_CASENO;

Begin

  For EachCase In CaseList
  Loop
    aRecord := DataSource || ',' ||
               EachCase.CSM_CASENO || ',' ||
               'DUP,' ||
               To_Char( EachCase.CSM_ISSUED_DATE, 'MM/DD/YY' ) || ',' ||
               EachCase.PRC_PARCEL_NO || ',' ||
               NVL( EachCase.CSM_NO_OF_UNITS, 2 ) || ',' ||
               '0,,';  -- Non-Residential SqFt, Owner, Address

    Utl_File.Put_Line( DatFileP, aRecord );
    howMany := howMany + 1;
  End Loop;

  Utl_File.Put_Line( LogFileP, '-> DUP : ' || To_Char( howMany ) );

  Exception
    When Others Then
    ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
    Utl_File.Put_Line( ErrFileP, 'Error: Procedure Extract_DUPs' );
```

```
End Extract_DUPs;


/*
--------------
Extract_MHPs()
--------------
*/
Procedure Extract_MHPs
(
  PermitType CASEMAIN.CASE_Type%Type
)

Is

  aRecord Varchar2( 255 );
  howMany Number := 0;

  Cursor CaseList Is

    Select M.CSM_CASENO, M.CSM_ISSUED_DATE, M.PRC_PARCEL_NO, E.CSM_NO_OF_UNITS
    From CASEMAIN M, CASE_EXTENDED E
    Where M.CSM_CASENO = E.CSM_CASENO
    And M.CASE_TYPE = PermitType
    And M.CSM_STATUS In ( 'APR', 'FNL' )
    And E.CSM_DEV_TYPE = 'MHP'
    And M.CSM_CASENO Not Like '%TEST%'
    And M.CSM_CASENO Not Like '%1900-%'
    And M.CSM_ISSUED_DATE Between sDate And eDate
    Order By M.CSM_CASENO;

Begin

  For EachCase In CaseList
  Loop
    aRecord := DataSource || ',' ||
               EachCase.CSM_CASENO || ',' ||
               'MHP,' ||
               To_Char( EachCase.CSM_ISSUED_DATE, 'MM/DD/YY' ) || ',' ||
               EachCase.PRC_PARCEL_NO || ',' ||
               NVL( EachCase.CSM_NO_OF_UNITS, 0 ) || ',' ||
               '0,,';  -- Non-Residential SqFt, Owner, Address

    Utl_File.Put_Line( DatFileP, aRecord );
    howMany := howMany + 1;
  End Loop;

  Utl_File.Put_Line( LogFileP, '-> MHP : ' || To_Char( howMany ) );

  Exception
    When Others Then
    ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
    Utl_File.Put_Line( ErrFileP, 'Error: Procedure Extract_MHPs' );

End Extract_MHPs;


/*
--------------
```

```
Extract_COMs()
--------------
*/
Procedure Extract_COMs
(
  PermitType CASEMAIN.CASE_TYPE%Type,
  CensusNo CASE_EXTENDED.CSM_CENSUS_CATEGORY%Type
)

Is

  aRecord Varchar2( 255 );
  howMany Number := 0;

  Cursor CaseList Is

    Select C.CSM_CASENO, M.CSM_ISSUED_DATE, M.PRC_PARCEL_NO, E.CSM_NO_OF_UNITS
    From CASE_COM C, CASEMAIN M, CASE_EXTENDED E
    Where C.CSM_CASENO = M.CSM_CASENO
    And M.CSM_CASENO = E.CSM_CASENO
    And E.CSM_CENSUS_CATEGORY = CensusNo
    And E.CSM_SCOPE_OF_WORK = 'NEW'
    And M.CSM_STATUS In ( 'APR', 'FNL' )
    And M.CSM_CASENO Not Like '%TEST%'
    And M.CSM_CASENO Not Like '%1900-%'
    And M.CSM_ISSUED_DATE Between sDate And eDate
    Order By M.CSM_CASENO;

Begin

  For EachCase In CaseList
  Loop
    aRecord := DataSource || ',' ||
               EachCase.CSM_CASENO || ',' ||
               PermitType || ',' ||
               To_Char( EachCase.CSM_ISSUED_DATE, 'MM/DD/YY' ) || ',' ||
               EachCase.PRC_PARCEL_NO || ',' ||
               NVL( EachCase.CSM_NO_OF_UNITS, 0 ) || ',' ||
               Bldg_Sqft( EachCase.CSM_CASENO ) ||
               ',,';  -- Owner, Address

    Utl_File.Put_Line( DatFileP, aRecord );
    howMany := howMany + 1;
  End Loop;

  Utl_File.Put_Line( LogFileP, '-> ' || PermitType || '-' || CensusNo || ' : '
|| To_Char( howMany ) );

  Exception
    When Others Then
    ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
    Utl_File.Put_Line( ErrFileP, 'Error: Procedure Extract_COMs' );

End Extract_COMs;


/*
-----------
Bldg_SqFt()
-----------
```

```
*/
Function Bldg_SqFt
(
  CaseNo CASEMAIN.CSM_CASENO%Type
)

Return Number

Is

  SqFt Number := 0;

Begin

  Select SUM( CSV_ITEM_AMT ) Into SqFt
  From CASE_VALU
  Where CSM_CASENO = CaseNo;

  Return SqFt;

  Exception
    When No_Data_Found Then
    Return SqFt;

End Bldg_SqFt;


/*
-----------------
Set_Date_Params()
-----------------
*/
Procedure Set_Date_Params
(
  sDate Out Date,
  eDate Out Date
)

Is

Begin

  sDate := Trunc( Add_Months( Sysdate, -1 ), 'MM' );
  eDate := Last_Day( Trunc( Add_Months( Sysdate, -1 ) ) );

End Set_Date_Params;


/*
-----------
Import_Data
-----------
*/
Procedure Import_Data

Is

Begin

  Load_Data( 'BG' );
  Load_Data( 'CAM' );
  Load_Data( 'CC' );
  Load_Data( 'LAC' );
  Load_Data( 'RDG' );
  Load_Data( 'VAN' );
  Load_Data( 'WAS' );

End Import_Data;


/*
---------
Load_Data
---------
*/
Procedure Load_Data
(
  DataSource PARCEL.PRC_JURISDICTION%Type
)

Is

  permitR rPermit;
  aRecord Varchar( 255 );
  permitCounter Number := 0;
  peopleCounter Number := 0;

Begin

  DBMS_OutPUT.ENABLE( 100000000 );

  DatFileN := DataSource || '.csv';
  DatFileP := Utl_File.Fopen( DirN, DatFileN, 'r' );
  LogFileN := DataSource || '_Import_' || rDate || '.log';
  LogFileP := Utl_File.Fopen( DirN, LogFileN, 'a' );
  ErrFileN := DataSource || '_Import_' || rDate || '.err';

  Utl_File.Put_Line( LogFileP, To_Char( Sysdate, 'HH24:Mi:SS' ) );
  -- Throw away the Heading
  Utl_File.Get_Line( DatFileP, aRecord );

  Begin
    Loop
      Utl_File.Get_Line( DatFileP, aRecord );
      Init_Permit( permitR );
      Read_Permit( DataSource, permitR, aRecord );

      If( Insert_Permit( permitR ) = True ) Then
        permitCounter := permitCounter + 1;
      End If;

      If( Insert_People( permitR ) = True ) Then
        peopleCounter := peopleCounter + 1;
      End If;

    End Loop;

    Exception
      When No_Data_Found Then
      Commit;
      Utl_File.Put_Line( LogFileP, '-> No of permits loaded : ' || To_Char(
permitCounter ) );
      Utl_File.Put_Line( LogFileP, '-> No of own/add loaded : ' || To_Char(
peopleCounter ) );
      Utl_File.Fclose( DatFileP );
      Utl_File.Fclose( LogFileP );
      Utl_File.Fclose( ErrFileP );

      When Others Then
      Rollback;
      ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
      Utl_File.Put_Line( ErrFileP, 'Error: Procedure Load_Data' );
      Utl_File.Fclose( DatFileP );
      Utl_File.Fclose( LogFileP );
      Utl_File.Fclose( ErrFileP );
  End;

End Load_Data;


/*
-------------
Init_Permit()
-------------
*/
Procedure Init_Permit
(
  permitR Out rPermit
)

Is

Begin

  permitR.PermitNo := Null;
  permitR.PermitType := Null;
  permitR.DateIssued := Null;
  permitR.ParcelNo := Null;
  permitR.NoUnits := 0;
  permitR.BldgSqFt := 0;

End Init_Permit;


/*
-------------
Read_Permit()
-------------
*/
Procedure Read_Permit
(
  DataSource PARCEL.PRC_JURISDICTION%Type,
  permitR Out rPermit,
  aRecord In Varchar2
)

Is

  positionR rPosition;
```

```
Begin

  Set_Position( DataSource, positionR );

  permitR.DataSource := DataSource;
  permitR.PermitNo := Parse_Data_By_Comma( aRecord, positionR.PermitNo_p );
  permitR.PermitType := Translate_PermitType( aRecord, positionR );
  permitR.DateIssued := To_Date( Parse_Data_By_Comma( aRecord,
positionR.DateIssued_p ), 'MM/DD/YY' );
  permitR.ParcelNo := Translate_ParcelNo( aRecord, positionR );
  permitR.NoUnits := To_Number( Parse_Data_By_Comma( aRecord,
positionR.NoUnits_p ) );
  permitR.BldgSqFt := To_Number( Parse_Data_By_Comma( aRecord,
positionR.BldgSqFt_p ) );
  permitR.Name := Substr( ( Parse_Data_By_Comma( aRecord, positionR.Name_p ) ),
1, 30 );
  permitR.Address := Substr( ( Parse_Data_By_Comma( aRecord,
positionR.Address_p ) ), 1, 30 );

End Read_Permit;


/*
--------------
Set_Position()
--------------
*/
Procedure Set_Position
(
  DataSource In PARCEL.PRC_JURISDICTION%Type,
  positionR Out rPosition
)

Is

Begin

  If DataSource In ( 'CC', 'VAN' ) Then
    positionR.PermitNo_p := 2;
    positionR.PermitType_p := 3;
    positionR.DateIssued_p := 4;
    positionR.ParcelNo_p := 5;
    positionR.NoUnits_p := 6;
    positionR.BldgSqFt_p := 7;
    positionR.Name_p := 8;
    positionR.Address_p := 9;
  Else
    positionR.PermitNo_p := 2;
    positionR.PermitType_p := 3;
    positionR.DateIssued_p := 4;
    positionR.ParcelNo_p := 5;
    positionR.NoUnits_p := 6;
    positionR.BldgSqFt_p := 7;
    positionR.Name_p := 8;
    positionR.Address_p := 9;
  End If;

End Set_Position;
```

```
/*
--------------------
Translate_ParcelNo()
--------------------
*/
Function Translate_ParcelNo
(
  aRecord In Varchar2,
  positionR In rPosition
)

Return Varchar2

Is

  rawText Varchar2( 255 );

Begin

  rawText := Parse_Data_By_Comma( aRecord, positionR.ParcelNo_p );
  Return Rpad( Lpad( Replace( rawText, '-', '' ), 9, '0' ), 10, '0' );

End Translate_ParcelNo;


/*
----------------------
Translate_PermitType()
----------------------
*/
Function Translate_PermitType
(
  aRecord In Varchar2,
  positionR In rPosition
)

Return Varchar2

Is

  rawText Varchar2( 255 );
  tmCode Varchar2( 3 ) := Null;

Begin

  rawText := Parse_Data_By_Comma( aRecord, positionR.PermitType_p );

  Select TM_CODE Into tmCode
  From GMA_TRANSLATOR
  Where ITEM = 'PERMIT_TYPE'
  And GMA_TEXT = Upper( rawText );

  Return( tmCode );

  Exception
    When Others Then
    Return( tmCode );

End Translate_PermitType;
```

```
/*
---------------------
Parse_Data_By_Comma()
---------------------
*/
Function Parse_Data_By_Comma
(
  aRecord In Varchar2,
  fieldNo In Number
)

Return Varchar2

Is

  aStrIng Varchar2( 80 );
  sComma Number;
  eComma Number;

Begin

  If fieldNo = 1 Then
    sComma := 0;
  Else
    sComma := InSTR( aRecord, ',', 1, fieldNo-1 );
  End If;

  eComma := InSTR( aRecord, ',', 1, fieldNo );

  If eComma <= 0 Then
    aStrIng := SUBSTR( aRecord, sComma+1 );
  Else
    aStrIng := SUBSTR( aRecord, sComma+1, eComma-sComma-1 );
  End If;

  Return Trim( aStrIng );

End Parse_Data_By_Comma;


/*
---------------
Insert_Permit()
---------------
*/
Function Insert_Permit
(
  permitR In rPermit
)

Return Boolean

Is

Begin

  Insert Into GMA_MAIN
  (
    DATA_SOURCE,
    PERMIT_NO,
    PRC_PARCEL_NO,
    PRC_AVP_NO,
    PERMIT_TYPE,
    DATE_ISSUED,
    NO_OF_UNITS,
    BLDG_SQFT,
    GMA_UPDATED
  )
  Values
  (
    permitR.DataSource,
    permitR.PermitNo,
    permitR.ParcelNo,
    0,
    permitR.PermitType,
    permitR.DateIssued,
    permitR.NoUnits,
    permitR.BldgSqFt,
    Sysdate
  );

  Return( True );

  Exception
    When Dup_Val_On_Index Then
      Utl_File.Put_Line( ErrFileP, 'Duplicate Permit: ' || permitR.PermitNo );

    When Others Then
      ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
      Utl_File.Put_Line( ErrFileP, 'Error: Procedure Insert_Permit>' ||
                                   ' PermitNo:' || permitR.PermitNo ||
                                   ' PermitType: ' || permitR.PermitType ||
                                   ' ParcelNo:' || permitR.ParcelNo
                       );
      Utl_File.Fclose( ErrFileP );
      Return( False );

End Insert_Permit;


/*
---------------
Insert_People()
---------------
*/
Function Insert_People
(
  permitR In rPermit
)

Return Boolean

Is

Begin

  If( permitR.Name Is Null And permitR.Address Is Null ) Then
    Return( False );
  End If;
  Insert Into GMA_PEOPLE
  (
    DATA_SOURCE,
    PERMIT_NO,
    NAME,
    ADDRESS
  )
  Values
  (
    permitR.DataSource,
    permitR.PermitNo,
    permitR.Name,
    permitR.Address
  );

  Return( True );

  Exception
    When Dup_Val_On_Index Then
      Utl_File.Put_Line( ErrFileP, 'Duplicate People: ' || permitR.PermitNo );

    When Others Then
      ErrFileP := Utl_File.Fopen( DirN, ErrFileN, 'a' );
      Utl_File.Put_Line( ErrFileP, 'Error: Procedure Insert_People>' ||
                                   ' PermitNo:' || permitR.PermitNo ||
                                   ' PermitType: ' || permitR.PermitType ||
                                   ' ParcelNo:' || permitR.ParcelNo
                       );
      Utl_File.Fclose( ErrFileP );
      Return( False );

End Insert_People;


/*
-------------------
Populate_Gis_Data()
-------------------
*/
Procedure Populate_Gis_Data

Is

  gd GIS.rGisData;

  -- Select all the parcels that need to be updated
  Cursor gParcels Is
    Select M.PRC_PARCEL_NO
    From GMA_MAIN M, PARCEL P
    Where M.PRC_PARCEL_NO = P.PRC_PARCEL_NO
    And M.PRC_AVP_NO = P.PRC_AVP_NO
    And
    ( P.PRC_URB_GRWTH_BNDRY Is Null
      Or P.PRC_JURISDICTION Is Null
      Or PRC_GIS_UPDATED Is Null
    );

Begin
```

```
  LogFileN := 'Gis_Data_' || rDate || '.log';
  LogFileP := Utl_File.Fopen( DirN, LogFileN, 'a' );
  ErrFileN := 'Gis_Data_' || rDate || '.err';

  For gParcel In gParcels
  Loop
    Begin
      If( Get_Gis_Data( gParcel.PRC_PARCEL_NO, gd ) = True ) Then
        GIS.pUpdate_Parcel_Table( LogFileP, gParcel.PRC_PARCEL_NO, gd );
        Commit;
      End If;

      Exception
        When Others Then
        Null;
    End;
  End Loop;

  Utl_File.Fclose( LogFileP );
  Utl_File.Fclose( ErrFileP );

  Exception
    When Others Then
    Utl_File.Fclose( LogFileP );
    Utl_File.Fclose( ErrFileP );

End Populate_Gis_Data;


/*
--------------
Get_Gis_Data()
--------------
*/
Function Get_Gis_Data
(
  ParcelNo In CASEMAIN.PRC_PARCEL_NO%Type,
  gd In Out GIS.rGisData
)

Return Boolean

Is

  domDocument XMLDOM.DOMDocument;
  domNode XMLDOM.DOMNode;
  eInvalidParcel Exception;
  eXmlParseError Exception;

Begin

  GIS.pWrite_Start( LogFileP );
  GIS.pWrite_Msg( 3, LogFileP, 'Enter <<<pGet_Gis_Data>>>' );
  GIS.pWrite_Msg( 0, LogFileP, 'ParcelNo: ' || ParcelNo );

  If( GIS.fIs_Valid_Parcel( ParcelNo ) ) = False THEN
    RAISE eInvalidParcel;
  End If;
  If GIS.fGet_Xml_Dom_Document( SUBSTR( ParcelNo, 1, 9 ), domDocument ) = False
THEN
    RAISE eXmlParseError;
  End If;

  -- convert the document object to a node
  domNode := XMLDOM.makeNode( domDocument );

  -- parse the DOM tree
  GIS.pParse_Dom_Tree( LogFileP, domNode, gd );

  GIS.pWrite_Msg( 3, LogFileP, 'Exit <<<Get_Gis_Data>>>' );
  GIS.pWrite_End( LogFileP );

  XMLDOM.freeDocument( domDocument );
  Return( True );

  Exception
    When eInvalidParcel THEN
      GIS.pWrite_Msg( 0, LogFileP, 'eInvalidParcel: ' || ParcelNo );
      GIS.pWrite_End( LogFileP );
      XMLDOM.freeDocument( domDocument );
      Return( False );

    When eXmlParseError THEN
      GIS.pWrite_Msg( 0, LogFileP, 'eXmlParseError: ' || ParcelNo );
      GIS.pWrite_End( LogFileP );
      XMLDOM.freeDocument( domDocument );
      Return( False );

    When Others THEN
      GIS.pWrite_Msg( 0, LogFileP, 'Others: ' || ParcelNo );
      GIS.pWrite_End( LogFileP );
      XMLDOM.freeDocument( domDocument );
      Return( False );

End Get_Gis_Data;


End GMA;
/
```

```
CREATE OR REPLACE PACKAGE GIS
IS

-- Package Specification
/*
 +-----------------------------------------------------------------------------
+
 |
|
 |                Real-Time Interface to GIS from Tidemark
|
 |
|
 |
|
 |                    Package Name  :  GIS
|
 |                         Version  :  2.0
|
 |                    Release Date  :  06/14/2005
|
 |                        Database  :  Oracle 8.1.7 (Release 3)
|
 |                 Operating System :  HP-UX
|
 |                          Author  :  Subu Swayam
|
 |                      Department  :  Information Services,
|
 |                                     Clark County, WA
|
 |
|
 |  Purpose
|
 |   To obtain Parcel data from GIS in real-time and populate Tidemark
|
 |   Database. This is done automatically when adding permit information
|
 |   Obtained from various Clark County jurisdictions
|
 |
|
 +-----------------------------------------------------------------------------
+
*/


  -- variables related to log files, error files and temp files
  DirN VARCHAR2( 25 );
  logFileN VARCHAR2( 25 );
  errFileN VARCHAR2( 25 );
  tmpFileN VARCHAR2( 25 );

  LOG_FILE_SUFFIX CONSTANT VARCHAR2( 4 ) := '.log';
  ERR_FILE_SUFFIX CONSTANT VARCHAR2( 4 ) := '.err';
  TMP_FILE_SUFFIX CONSTANT VARCHAR2( 7 ) := '.prc';

  -- MLQ Server
  MLQ_PREFIX CONSTANT VARCHAR2( 8 ) := 'http://';
  MLQ_SUFFIX CONSTANT VARCHAR2( 20 ) := '/MLQ2/MyMLQ.aspx?SN=';
  MLQ_PSERVER CONSTANT VARCHAR2( 4 ) := 'nt04';
  MLQ_SSERVER CONSTANT VARCHAR2( 4 ) := 'nt79';

  -- Time Tracking Start
  GIS_START DATE;
  MLQ_START DATE;
  MLQ_END DATE;
  -- Time Tracking End

  -- how much info to be logged?
  -- 0 = None, 1 = Low, 2 = Medium, 3 = High
  MSG_LOG_LEVEL CONSTANT NUMBER := 3;

  -- Number of days to accept the previously copied GIS data
  -- in case of a failure to connect to GIS
  ACCEPTABLE_DAYS CONSTANT NUMBER := 45;
  MAX_BUFFER_SIZE CONSTANT NUMBER := 255;
  MULTIPLE_DATA CONSTANT VARCHAR2( 3 ) := '***';
  BLANK_DATA CONSTANT VARCHAR2( 1 ) := '~';
  NO_TM_CODE CONSTANT VARCHAR2( 1 ) := '#';
  DUMMY_PARCEL_NUMBER CONSTANT PARCEL.PRC_PARCEL_NO%TYPE := 'DummyParcel';

  -- XML tags
  ARCH_PREDICT_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Amodsim/PROBABILIT';
  ASSR_SQFT_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Taxlots/ASSR_SQFT';
  CRITICAL_AREA_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Critical_Area/SQFT';
  SCM_AREA_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/salmor/AREADESC';
  CARA_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Aquifer/AQUIFER';
  CR_GORGE_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Crgnsa/LOCATION';
  COMPLAN_DESIGN_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Complan/ABBREV';
  FIRE_DISTRICT_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Firedst/FIREDST';
  FLOOD_ZONE_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Flood/FLOOD';
  GIS_SQFT_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Taxlots/GIS_SQFT';
  INSPECTOR_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/inspect/INSPECT_ID';
  JURISDICTION_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Cities/NAME';
  NEIGHBORHOOD_TAG CONSTANT VARCHAR2( 30 ) := 'MLQ/Nbrassoc/NBRASSOC';
  PARK_DISTRICT_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Parkdst/PARKDST';
--  QUARTER_SECTION_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/qsection/QUARTER';
  SCHOOL_DISTRICT_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Schdst/SCHDST';
  SEWER_DISTRICT_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Sewerdst/DESC';
  TRAFFIC_DISTRICT_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Tif/NAME';
  URBAN_GROWTH_AREA_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/ugabnd/DESC';
  WATER_DISTRICT_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Waterdst/WATERDST_D';
  WILDLAND_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Wildland/WILDLAND';
  ZONING_TAG CONSTANT VARCHAR2( 30 ) := '/MLQ/Zoning/ABBREV';

  -- Global Variables
  gisText VARCHAR2( 255 ) := NULL;
  howMany NUMBER := 0;
  -- Is the program running in the county or in the city instance
  isCounty BOOLEAN := FALSE;
  -- Name of the Windows machine where MLQ is residing
  winServer VARCHAR2( 4 ) := 'none';

  --  Parcel Tag Variables
  -- Salmon Creek Moratorium
/*
  SCM_ATTR_CODE CONSTANT ATTRIB_LIB.ATTR_CODE%TYPE := 'SCMT';
  SCM_ATTR_ORDER CONSTANT ATTRIB_LIB.ATTR_ORDERING%TYPE := 197;
*/
  -- Rural Land Division Moratorium
  RLDM_ATTR_CODE CONSTANT ATTRIB_LIB.ATTR_CODE%TYPE := 'RLDM';
  RLDM_ATTR_ORDER CONSTANT ATTRIB_LIB.ATTR_ORDERING%TYPE := 198;

  -- Columbia River Gorge NSA
  CRG_ATTR_CODE CONSTANT ATTRIB_LIB.ATTR_CODE%TYPE := 'GORG';
  CRG_ATTR_ORDER CONSTANT ATTRIB_LIB.ATTR_ORDERING%TYPE := 222;

  -- GIS Record Structure to hold GIS data
  TYPE rGISDATA is RECORD (
    archPredict CASE_EXTENDED.CSM_ARCH_PREDICT%TYPE,
    cara CASE_EXTENDED.CSM_CARA%TYPE,
    compPlanDesign CASE_EXTENDED.CSM_COMP_PLAN_DESIGN%TYPE,
    criticalArea CASE_EXTENDED.CSM_CRITICAL_AREA_SF%TYPE,
    floodZone CASE_EXTENDED.CSM_FLOOD_ZONE%TYPE,
    gisArea PARCEL.PRC_GIS_AREA_SF%TYPE,
    inspector CASE_EXTENDED.CSM_INSPECTOR%TYPE,
    jurisdiction CASE_EXTENDED.CSM_JURISDICTION%TYPE,
    m_neighborhood CASE_EXTENDED.CSM_MULT_NGHBRHD%TYPE,
    neighborhood CASE_EXTENDED.CSM_NEIGHBORHOOD%TYPE,
    fireDistrict CASE_EXTENDED.CSM_FIRE_DISTRICT%TYPE,
    parkDistrict CASE_EXTENDED.CSM_PARK_DISTRICT%TYPE,
    schoolDistrict CASE_EXTENDED.CSM_SCHOOL_DISTRICT%TYPE,
    p_schoolDistrict PARCEL.PRC_SCHOOL_DIST%TYPE,
    sewerDistrict CASE_EXTENDED.CSM_SWR_DISTRICT%TYPE,
    trafficDistrict CASE_EXTENDED.CSM_TRAFFIC_DISTRICT%TYPE,
    waterDistrict CASE_EXTENDED.CSM_WTR_DISTRICT%TYPE,
    urbanGrowthArea CASE_EXTENDED.CSM_URB_GRWTH_BNDRY%TYPE,
    wildLand CASE_EXTENDED.CSM_WILDLAND%TYPE,
    zoning1 CASE_EXTENDED.CSM_ZONING1%TYPE,
    zoning2 CASE_EXTENDED.CSM_ZONING2%TYPE
--    quarterNo PARCEL.PRC_QUARTER%TYPE,
--    sectionNo PARCEL.PRC_SECTION%TYPE,
--    townshipNo PARCEL.PRC_TOWNSHIP%TYPE,
--    rangeNo PARCEL.PRC_RANGE%TYPE,
--    soilType CASE_EXTENDED.CSM_SOIL_CLASS%TYPE,
--    slope CASE_EXTENDED.CSM_SLOPE%TYPE,
--    archSiteBuffers CASE_EXTENDED.CSM_ARCH_BUFFER%TYPE,
--    yearBuilt PARCEL.PRC_PRI_CONST%TYPE,
  );


/*
---------------
pGet_Gis_Data()
---------------
  -- INPUT(s) : Case Number
  -- OUTPUT(s) : GISDATA Record
  -- CALLER(s) : GIS_CASE_EXTENDED Trigger
  -- PURPOSE : When a new case is created, a new record is added to the
               Case Extended table which triggers GIS_CASE_EXTENDED and
                     pGet_Gis_Data() gets called.
*/
  PROCEDURE pGet_Gis_Data
  (
    caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
    gd OUT rGISDATA
  );


/*
------------------
```

```
pUpdate_Gis_Data()
------------------
  -- INPUT(s) : Case Number
  -- OUTPUT(s) : None
  -- CALLER(s) : GIS_CASE_ACTION Trigger
  -- PURPOSE : When a new activity is added and signed-off, a new record is
               added to the Case Action table which triggers GIS_CASE_ACTION
                     and pUpdate_Gis_Data() gets called.
*/
  PROCEDURE pUpdate_Gis_Data
  (
    caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE
  );


/*
--------------------
fGet_Parcel_Number()
--------------------
  -- INPUT(s) : Case Number
  -- OUTPUT(s) : Parcel Number
  -- CALLER(s) : pUpdate_Gis_Data()
  -- PURPOSE : Obtains Parcel Number from the Case Main table; case must be
               existing to obtain it!  If successful returns TRUE else FALSE.
*/
  FUNCTION fGet_Parcel_Number
  (
    caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
    parcelNo OUT PARCEL.PRC_PARCEL_NO%TYPE
  )
  RETURN BOOLEAN;


/*
-----------------------
fGet_Xml_Dom_Document()
-----------------------
  -- INPUT(s) : Parcel Number
  -- OUTPUT(s) : DOM Document
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : First calls the primary MLQ Server. If fails, calls
               the secondary MLQ Server. Also keeps track of response time.
*/
  FUNCTION fGet_Xml_Dom_Document
  (
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
    domDocument OUT XMLDOM.DOMDocument
  )
  RETURN BOOLEAN;


/*
--------------
fExecute_MLQ()
--------------
  -- INPUT(s) : URL where MLQ Server resides
  -- OUTPUT(s) : DOM Document
  -- CALLER(s) : fGet_Xml_Dom_Document()
  -- PURPOSE : Connects to MLQ Server, obtains XML Document and generates
               DOM Document
*/
```

```
  FUNCTION fExecute_MLQ
  (
    xmlurl VARCHAR2,
    domDocument OUT XMLDOM.DOMDocument
  )
  RETURN BOOLEAN;

/*
-----------------
pParse_Dom_Tree()
-----------------
  -- INPUT(s) : DOM Node Object
  -- OUTPUT(s) : GISDATA Record
  -- CALLER(s) : pUpdate_Gis_Data()
  -- PURPOSE : Parses DOM Node and copies various GIS data into GISDATA Record
*/
  PROCEDURE pParse_Dom_Tree
  (
    logFileP IN UTL_FILE.file_type,
    domNode XMLDOM.DOMNode,
    gd OUT rGISDATA
  );


/*
-----------------------------
pUpdate_Case_Extended_Table()
-----------------------------
  -- INPUT(s) : Case Number, GISDATA Record
  -- OUTPUT(s) : None
  -- CALLER(s) : pUpdate_Gis_Data()
  -- PURPOSE : Updates Case Extended table with the GIS data.
*/
  PROCEDURE pUpdate_Case_Extended_Table
  (
    logFileP IN UTL_FILE.file_type,
    caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
    gd IN rGISDATA
  );


/*
----------------------
pUpdate_Parcel_Table()
----------------------
  -- INPUT(s) : Parcel Number, GISDATA Record
  -- OUTPUT(s) : None
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : Updates Parcel table with the GIS data. This data will be
               used to update cases on this parcel in the future within a
               "reasonable" period of time in case of failure to connect
               to GIS system.
*/
  PROCEDURE pUpdate_Parcel_Table
  (
    logFileP IN UTL_FILE.file_type,
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
    gd IN rGISDATA
  );
```

```
/*
------------
pWrite_Msg()
------------
  -- INPUT(s) : Log (Debug) level, File pointer, Message
  -- OUTPUT(s) : None
  -- CALLER(s) : Various Procedures and Functions
  -- PURPOSE : Logs messages to file pointed by the File pointer based on the
               Log level specified.  The contents of this file could be
               helpful for debugging.
*/
  PROCEDURE pWrite_Msg
  (
    logLevel IN NUMBER := 3,
    fp IN UTL_FILE.file_type,
    message IN VARCHAR2
  );


/*
------------------------
pParse_Quarter_Section()
------------------------
  -- INPUT(s) : Quarter Section String
  -- OUTPUT(s) : GISDATA Record with Quarter Section Data
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses the Quarter Section string and identifies Township,
               Range, Section and Quarter
*/
/*
  PROCEDURE pParse_Quarter_Section
  (
    quarterSection IN VARCHAR2,
    gd IN OUT rGISDATA
  );
*/

/*
--------------------
fGet_Tidemark_Code()
--------------------
  -- INPUT(s) : GIS String
  -- OUTPUT(s) : Tidemark Code
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Looks up GIS_TM_CODES table and translates certain data
               received from the GIS to codes used by Tidemark.
*/
  FUNCTION fGet_Tidemark_Code
  (
    gItem GIS_TRANSLATOR.ITEM%TYPE,
    gisText GIS_TRANSLATOR.GIS_TEXT%TYPE
  )
  RETURN VARCHAR2;


/*
-----------------
fGet_Total_Tags()
-----------------
  -- INPUT(s) : DOM Node Object, XML Tag
  -- OUTPUT(s) : Count
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : GIS can have multiple data values for a given Layer/Field.
               Counts and returns the number of values for the given tag.
*/
  FUNCTION fGet_Total_Tags
  (
    domNode IN XMLDOM.DOMNode,
    tag IN VARCHAR2
  )
  RETURN NUMBER;


/*
----------------
fGet_Tag_Value()
----------------
  -- INPUT(s) : DOM Node Object, XML Tag, Tag Order (default = 1)
  -- OUTPUT(s) : Data Value
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : GIS can have multiple data values for a given Layer/Field.
               Counts and returns the number of values for the given tag.
*/
  FUNCTION fGet_Tag_Value
  (
    domNode IN XMLDOM.DOMNode,
    tag IN VARCHAR2,
      nTag IN NUMBER := 1
  )
  RETURN VARCHAR2;


/*
--------------
pFatal_Error()
--------------
  -- INPUT(s) : Error Type
  -- OUTPUT(s) : None
  -- CALLER(s) : Various Procedures and Functions
  -- PURPOSE : When an Exception occurs during the execution of the package,
               this procedure creates an error file - caseNo.err and logs the
               error messages in the log file which could be used for
debugging.
*/
  PROCEDURE pFatal_Error
  (
    errFileN IN VARCHAR2,
    logFileP IN UTL_FILE.file_type,
    errType IN GIS_ERR_MSG.ERR_TYPE%TYPE
  );


/*
------------------
fIs_Valid_Parcel()
------------------
  -- INPUT(s) : Parcel Number
  -- OUTPUT(s) : TRUE / FALSE
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : Checks to validate the Parcel Number before attempting
               to connect to GIS to obtain data.
*/
```

```
  FUNCTION fIs_Valid_Parcel
  (
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE
  )
  RETURN BOOLEAN;


/*
----------------------
pWrite_Parcel_Number()
----------------------
  -- INPUT(s) : Case Number, Parcel Number
  -- OUTPUT(s) : None
  -- CALLER(s) : GIS_CASE_MAIN Trigger
  -- PURPOSE : Writes the Parcel Number to a temporary file (caseNo.prc)
               in /v01/tc???/dump directory
*/
  PROCEDURE pWrite_Parcel_Number
  (
--    logFileP IN UTL_FILE.file_type,
    caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE
  );


/*
---------------------
fRead_Parcel_Number()
---------------------
  -- INPUT(s) : Case Number
  -- OUTPUT(s) : Parcel Number
  -- CALLER(s) : pGet_Gis_Data()
  -- PURPOSE : Reads the Parcel Number from the temporary file (caseNo.prc)
               and returns TRUE if valid or FALSE if invalid.
*/
  FUNCTION fRead_Parcel_Number
  (
    caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
    parcelNo OUT PARCEL.PRC_PARCEL_NO%TYPE
  )
  RETURN BOOLEAN;


/*
-------------------------
fCopy_Local_Parcel_Data()
-------------------------
  -- INPUT(s) : Case Number, Parcel Number
  -- OUTPUT(s) : GISDATA Record
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : In case of failure to connect to the GIS system, the GIS
               data is copied from the Parcel table if the Parcel Data was
               updated by the GIS link within a "reasonable" period of time.
*/
  FUNCTION fCopy_Local_Parcel_Data
  (
    logFileP IN UTL_FILE.file_type,
    caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
    gd OUT rGISDATA
  )
  RETURN BOOLEAN;


/*
---------------------
fCopy_Arch_Predict()
---------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Quarter Section into GISDATA Record
*/
  PROCEDURE fCopy_Arch_Predict
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
-----------------------
fCopy_Quarter_Section()
-----------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Quarter Section into GISDATA Record
*/
/*
  PROCEDURE fCopy_Quarter_Section
  (
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );
*/

/*
------------------------
fCopy_Gis_Jurisdiction()
------------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Jurisdiction into GISDATA Record
*/
  PROCEDURE fCopy_Gis_Jurisdiction
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
---------------------------
fCopy_Parcel_Jurisdiction()
---------------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Jurisdiction into GISDATA Record
*/
  PROCEDURE fCopy_Parcel_Jurisdiction
  (
    logFileP IN UTL_FILE.file_type,
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
    gd IN OUT rGISDATA
  );


/*
--------------
fCopy_Zoning()
--------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Zoning(s) into GISDATA Record
*/
  PROCEDURE fCopy_Zoning
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
----------------------
fCopy_Complan_Design()
----------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Complan Design into GISDATA Record
*/
  PROCEDURE fCopy_Complan_Design
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
----------------------
fCopy_Critical_Area()
----------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Critical Area into GISDATA Record
*/
  PROCEDURE fCopy_Critical_Area
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );
```

```
/*
----------------
fCopy_Gis_Sqft()
----------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Critical Area into GISDATA Record
*/
  PROCEDURE fCopy_Gis_Sqft
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
--------------------
fCopy_Neighborhood()
--------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Neighborhood into GISDATA Record
*/
  PROCEDURE fCopy_Neighborhood
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
-----------------------
fCopy_School_District()
-----------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies School District into GISDATA Record
*/
  PROCEDURE fCopy_School_District
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
---------------------
fCopy_Fire_District()
---------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Fire District into GISDATA Record
*/
  PROCEDURE fCopy_Fire_District
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
------------------------
fCopy_Traffic_District()
------------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Traffic District into GISDATA Record
*/
  PROCEDURE fCopy_Traffic_District
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
---------------------
fCopy_Park_District()
---------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Park District into GISDATA Record
*/
  PROCEDURE fCopy_Park_District
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
----------------------
fCopy_Sewer_District()
----------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Sewer District into GISDATA Record
*/
  PROCEDURE fCopy_Sewer_District
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
----------------------
fCopy_Water_District()
```

```
----------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Water District into GISDATA Record
*/
  PROCEDURE fCopy_Water_District
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
----------------------
fCopy_Inspector_Area()
----------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Inspector Area into GISDATA Record
*/
  PROCEDURE fCopy_Inspector_Area
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
------------
fCopy_Cara()
------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Aquifer into GISDATA Record
*/
  PROCEDURE fCopy_Cara
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
------------------
fCopy_Flood_Zone()
------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Flood Zone into GISDATA Record
*/
  PROCEDURE fCopy_Flood_Zone
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
------------------
fCopy_Wildland()
------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies Wildland into GISDATA Record
*/
  PROCEDURE fCopy_Wildland
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
-------------------------
fCopy_Urban_Growth_Area()
-------------------------
  -- INPUT(s) : DOM Node Object, GISDATA Record
  -- OUTPUT(s) : Updated GISDATA Record
  -- CALLER(s) : pParse_Dom_Tree()
  -- PURPOSE : Parses DOM Node and copies UGA into GISDATA Record
*/
  PROCEDURE fCopy_Urban_Growth_Area
  (
    logFileP IN UTL_FILE.file_type,
    domNode IN XMLDOM.DOMNode,
    gd IN OUT rGISDATA
  );


/*
------------------------
pCheck_For_Parcel_Tags()
------------------------

  -- INPUT(s) : Parcel Number
  -- OUTPUT(s) : DOM Document
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : Checks for parcel tag entries in Parcel_Attrib table
               for Salmon Creek and Columbia Gorge Parcels
*/
  PROCEDURE pCheck_For_Parcel_Tags
  (
    logFileP IN UTL_FILE.file_type,
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
    domNode IN XMLDOM.DOMNode
  );


/*
-----------------
pAdd_Parcel_Tag()
-----------------
```

```
  -- INPUT(s) : Parcel Number
  -- OUTPUT(s) : DOM Document
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : Adds an entry in Parcel_Attrib table for the given parcel
               if it does not exists already
*/
  PROCEDURE pAdd_Parcel_Tag
  (
    logFileP IN UTL_FILE.file_type,
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
    attribCode IN ATTRIB_LIB.ATTR_CODE%TYPE,
    attribOrder IN ATTRIB_LIB.ATTR_ORDERING%TYPE
  );


/*
-----------------
pInsert_Gis_Log()
-----------------
  -- INPUT(s) : Case No, Parcel No, Action, Status and Comments
  -- OUTPUT(s) : None
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : Adds an entry in GIS_LOG table. Keeps track of who is using
               the system and how it is being used. The info can be viewed
               using GisStatus.rpt and can be submitted to upper Management!
*/
  PROCEDURE pInsert_Gis_Log
  (
    logFileP IN UTL_FILE.file_type,
    caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
    action IN GIS_LOG.GIS_ACTION%TYPE,
    status IN GIS_LOG.GIS_STATUS%TYPE,
    comments IN GIS_LOG.GIS_COMMENTS%TYPE := NULL
  );


/*
--------------
pParse_Error()
--------------
  -- INPUT(s) : Error Type, Case No, Parcel No
  -- OUTPUT(s) : Updated GISDATA Record (if local data is available)
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : Handles errors occurred during parsing the XML document
*/
  PROCEDURE pParse_Error
  (
    logFileP IN UTL_FILE.file_type,
    eType IN GIS_ERR_MSG.ERR_TYPE%TYPE,
    caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
    parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
    action IN GIS_LOG.GIS_ACTION%TYPE,
    gd OUT rGISDATA
  );


/*
--------------
pWrite_Start()
--------------
```

```
  -- INPUT(s) : None
  -- OUTPUT(s) : None
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : Writes header info in the log file
*/
  PROCEDURE pWrite_Start
  (
    logFileP IN UTL_FILE.file_type
  );


/*
------------
pWrite_End()
------------
  -- INPUT(s) : None
  -- OUTPUT(s) : None
  -- CALLER(s) : pGet_Gis_Data(), pUpdate_Gis_Data()
  -- PURPOSE : Writes footer info in the log file
*/
  PROCEDURE pWrite_End
  (
    logFileP IN UTL_FILE.file_type
  );


END GIS;
/
CREATE OR REPLACE PACKAGE BODY GIS
IS

-- GIS Package Body

/*
---------------
pGet_Gis_Data()
---------------
*/
PROCEDURE pGet_Gis_Data
(
  caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
  gd OUT rGISDATA
)

IS

  parcelNo PARCEL.PRC_PARCEL_NO%TYPE := NULL;
  domDocument XMLDOM.DOMDocument;
  domNode XMLDOM.DOMNode;
  logFileP UTL_FILE.file_type;
  action GIS_LOG.GIS_ACTION%TYPE := 'New';
  DB_INSTANCE VARCHAR2( 10 );
  eInvalidParcel EXCEPTION;
  eXmlParseError EXCEPTION;

BEGIN

  SELECT lower( NAME ) INTO DB_INSTANCE FROM SYS.v_$database;
  DirN := '/v01/' || DB_INSTANCE || '/dump/gislog';
  logFileN := caseNo || LOG_FILE_SUFFIX;
  errFileN := caseNo || ERR_FILE_SUFFIX;
  logFileP := utl_file.fopen( DirN, logFileN, 'a' );

  -- Is the program running in the county instance
  IF SUBSTR( DB_INSTANCE, 2, 1 ) = 'c' THEN
    isCounty := TRUE;
  END IF;

  pWrite_Start( logFileP );
  pWrite_Msg( 3, logFileP, 'Enter <<<pGet_Gis_Data>>>' );
  pWrite_Msg( 0, logFileP, 'CaseNo: ' || caseNo );

  IF fRead_Parcel_Number( caseNo, parcelNo ) = FALSE THEN
    RAISE eInvalidParcel;
  END IF;

  pWrite_Msg( 0, logFileP, 'ParcelNo: ' || parcelNo );

  IF fGet_Xml_Dom_Document( SUBSTR( parcelNo, 1, 9 ), domDocument ) = FALSE
THEN
    RAISE eXmlParseError;
  END IF;

  -- convert the document object to a node
  domNode := XMLDOM.makeNode( domDocument );

  -- parse the DOM tree
  pParse_Dom_Tree( logFileP, domNode, gd );

  pUpdate_Parcel_Table( logFileP, parcelNo, gd );

  -- Check for Parcel tags only for the County
  IF isCounty = TRUE THEN
    pCheck_For_Parcel_Tags( logFileP, parcelNo, domNode );
  END IF;

  pInsert_Gis_Log( logFileP, caseNo, parcelNo, 'New', 'Success' );
  pWrite_Msg( 3, logFileP, 'Exit <<<pGet_Gis_Data>>>' );
  pWrite_End( logFileP );

  UTL_FILE.fclose_all;
  XMLDOM.freeDocument( domDocument );

  EXCEPTION
    WHEN eInvalidParcel THEN
      pWrite_Msg( 0, logFileP, 'ParcelNo: ' || parcelNo );
      fCopy_Parcel_Jurisdiction( logFileP, parcelNo, gd );
      pInsert_Gis_Log( logFileP, caseNo, parcelNo, action, 'Failure', 'Invalid
Parcel' );
        pFatal_Error( errFileN, logFileP, 'eInvalidParcel' );
      pWrite_End( logFileP );
      UTL_FILE.fclose_all;

    WHEN eXmlParseError THEN
      pParse_Error( logFileP, 'eXmlParseError', caseNo, parcelNo, action, gd );
      pWrite_End( logFileP );
      UTL_FILE.fclose_all;
--      XMLDOM.freeDocument( domDocument );

    WHEN OTHERS THEN
      pParse_Error( logFileP, 'OthersGet', caseNo, parcelNo, action, gd );
      pWrite_End( logFileP );
      UTL_FILE.fclose_all;
      XMLDOM.freeDocument( domDocument );

END pGet_Gis_Data;


/*
------------------
pUpdate_Gis_Data()
------------------
*/
PROCEDURE pUpdate_Gis_Data
(
  caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE
)

IS

  parcelNo PARCEL.PRC_PARCEL_NO%TYPE := NULL;
  gd rGISDATA;
  domDocument XMLDOM.DOMDocument;
  domNode XMLDOM.DOMNode;
  logFileP UTL_FILE.file_type;
  action GIS_LOG.GIS_ACTION%TYPE := 'Upd';
  DB_INSTANCE VARCHAR2( 10 );
  eInvalidParcel EXCEPTION;
  eXmlParseError EXCEPTION;

BEGIN

  SELECT lower( NAME ) INTO DB_INSTANCE FROM SYS.v_$database;
  DirN := '/v01/' || DB_INSTANCE || '/dump/gislog';
  logFileN := caseNo || LOG_FILE_SUFFIX;
  errFileN := caseNo || ERR_FILE_SUFFIX;
  logFileP := utl_file.fopen( DirN, logFileN, 'a' );

  -- Is the program running in the county instance
  IF SUBSTR( DB_INSTANCE, 2, 1 ) = 'c' THEN
    isCounty := TRUE;
  END IF;

  pWrite_Start( logFileP );
  pWrite_Msg( 3, logFileP, 'Enter <<<pUpdate_Gis_Data>>>' );
  pWrite_Msg( 0, logFileP, 'CaseNo: ' || caseNo );

  IF fGet_Parcel_Number( caseNo, parcelNo ) = FALSE THEN
    RAISE eInvalidParcel;
  END IF;

  pWrite_Msg( 0, logFileP, 'ParcelNo: ' || parcelNo );

  IF fGet_Xml_Dom_Document( SUBSTR( parcelNo, 1, 9 ), domDocument ) = FALSE
THEN
    RAISE eXmlParseError;
  END IF;

  -- convert the document object to a node
  domNode := XMLDOM.makeNode( domDocument );

  -- parse the DOM tree
  pParse_Dom_Tree( logFileP, domNode, gd );
  pUpdate_Case_Extended_Table( logFileP, caseNo, gd );
  pUpdate_Parcel_Table( logFileP, parcelNo, gd );

  -- Check for Parcel tags only for the County
  IF isCounty = TRUE THEN
    pCheck_For_Parcel_Tags( logFileP, parcelNo, domNode );
  END IF;

  pInsert_Gis_Log( logFileP, caseNo, parcelNo, 'Upd', 'Success' );
  pWrite_Msg( 3, logFileP, 'Exit <<<pUpdate_Gis_Data>>>' );
  pWrite_End( logFileP );

  UTL_FILE.fclose_all;
  XMLDOM.freeDocument( domDocument );

  EXCEPTION
    WHEN eInvalidParcel THEN
      pWrite_Msg( 0, logFileP, 'ParcelNo: ' || parcelNo );
      fCopy_Parcel_Jurisdiction( logFileP, parcelNo, gd );
      pUpdate_Case_Extended_Table( logFileP, caseNo, gd );
      pInsert_Gis_Log( logFileP, caseNo, parcelNo, action, 'Failure', 'Invalid
Parcel' );
        pFatal_Error( errFileN, logFileP, 'eInvalidParcel' );
      pWrite_End( logFileP );
      UTL_FILE.fclose_all;

    WHEN eXmlParseError THEN
      pParse_Error( logFileP, 'eXmlParseError', caseNo, parcelNo, action, gd );
      pUpdate_Case_Extended_Table( logFileP, caseNo, gd );
      pWrite_End( logFileP );
      UTL_FILE.fclose_all;
--      XMLDOM.freeDocument( domDocument );

    WHEN OTHERS THEN
      pParse_Error( logFileP, 'OthersUpd', caseNo, parcelNo, action, gd );
      pUpdate_Case_Extended_Table( logFileP, caseNo, gd );
      pWrite_End( logFileP );
      UTL_FILE.fclose_all;
      XMLDOM.freeDocument( domDocument );

END pUpdate_Gis_Data;


/*
--------------------
fGet_Parcel_Number()
--------------------
*/
FUNCTION fGet_Parcel_Number
(
  caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
  parcelNo OUT PARCEL.PRC_PARCEL_NO%TYPE
)

RETURN BOOLEAN

IS

  eDummyParcel EXCEPTION;
```

```
BEGIN

  SELECT prc_parcel_no INTO parcelNo
  FROM casemain
  WHERE csm_caseno = caseNo;

  RETURN fIs_Valid_Parcel( parcelNo );

  EXCEPTION
    WHEN OTHERS THEN
      RETURN FALSE;

END fGet_Parcel_Number;


/*
-----------------------
fGet_Xml_Dom_Document()
-----------------------
*/
FUNCTION fGet_Xml_Dom_Document
(
  parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
  domDocument OUT XMLDOM.DOMDocument
)
RETURN BOOLEAN

IS

  xmlurl VARCHAR2( 100 );
  parser XMLPARSER.Parser;
  status BOOLEAN;

BEGIN

  winServer := MLQ_PSERVER;
  xmlurl := MLQ_PREFIX || MLQ_PSERVER || MLQ_SUFFIX || parcelNo;

  -- Time Tracking Start
  MLQ_START := SYSDATE;

  -- First run MLQ from the primary server.
  status := fExecute_MLQ( xmlurl, domDocument );

  -- If fails, run from the secondary server
  IF status = FALSE THEN
    winServer := MLQ_SSERVER;
    xmlurl := MLQ_PREFIX || MLQ_SSERVER || MLQ_SUFFIX || parcelNo;
    status := fExecute_MLQ( xmlurl, domDocument );
  END IF;

  -- Time Tracking End
  MLQ_END := SYSDATE;

  RETURN status;

END fGet_Xml_Dom_Document;


/*
--------------
```

```
fExecute_MLQ()
--------------
*/
FUNCTION fExecute_MLQ
(
  xmlurl VARCHAR2,
  domDocument OUT XMLDOM.DOMDocument
)
RETURN BOOLEAN

IS

  parser XMLPARSER.Parser;

BEGIN

  parser := XMLPARSER.newParser;

  -- parse the XML string and create a DOM tree and free the parser
  domDocument := XMLPARSER.parse( xmlurl );
  XMLPARSER.freeParser( parser );

  RETURN TRUE;

  EXCEPTION
    WHEN OTHERS THEN
      XMLPARSER.freeParser( parser );
      RETURN FALSE;

END fExecute_MLQ;


/*
-----------------
pParse_Dom_Tree()
-----------------
*/
PROCEDURE pParse_Dom_Tree
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 3, logFileP, 'Enter <<<pParse_Dom_Tree>>>' );
  pWrite_Msg( 1, logFileP, '==>Parsing XML Document ...' );

  fCopy_Arch_Predict( logFileP, domNode, gd );
  fCopy_Cara( logFileP, domNode, gd );
  fCopy_Complan_Design( logFileP, domNode, gd );
  fCopy_Critical_Area( logFileP, domNode, gd );
  fCopy_Gis_Sqft( logFileP, domNode, gd );
  fCopy_Fire_District( logFileP, domNode, gd );
  fCopy_Flood_Zone( logFileP, domNode, gd );
  IF isCounty = TRUE THEN
    fCopy_Inspector_Area( logFileP, domNode, gd );
  END IF;
  fCopy_Gis_Jurisdiction( logFileP, domNode, gd );
  fCopy_Neighborhood( logFileP, domNode, gd );
  fCopy_Park_District( logFileP, domNode, gd );
--  fCopy_Quarter_Section( domNode, gd );
  fCopy_Traffic_District( logFileP, domNode, gd );
  fCopy_School_District( logFileP, domNode, gd );
  fCopy_Sewer_District( logFileP, domNode, gd );
  fCopy_Urban_Growth_Area( logFileP, domNode, gd );
  fCopy_Water_District( logFileP, domNode, gd );
  fCopy_Wildland( logFileP, domNode, gd );
  fCopy_Zoning( logFileP, domNode, gd );
  -- Copy the Inspector Area only for the County!

  pWrite_Msg( 3, logFileP, 'Exit <<<pParse_Dom_Tree>>>' );

  EXCEPTION
    WHEN OTHERS THEN
      pWrite_Msg( 3, logFileP, '***Error <<<pParse_Dom_Tree>>>' );
      pFatal_Error( errFileN, logFileP, 'BadXMLtag' );

END pParse_Dom_Tree;


/*
-----------------------------
pUpdate_Case_Extended_Table()
-----------------------------
*/
PROCEDURE pUpdate_Case_Extended_Table
(
  logFileP IN UTL_FILE.file_type,
  caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
  gd IN rGISDATA
)

IS

BEGIN

  pWrite_Msg( 3, logFileP, 'Enter <<<pUpdate_Case_Extended_Table>>>' );
  update case_extended set
    CSM_ARCH_PREDICT = gd.archPredict,
    CSM_CARA = gd.cara,
      CSM_COMP_PLAN_DESIGN = gd.compPlanDesign,
    CSM_FLOOD_ZONE = gd.floodZone,
    CSM_INSPECTOR = gd.inspector,
    CSM_JURISDICTION = gd.jurisdiction,
    CSM_MULT_NGHBRHD = gd.m_neighborhood,
      CSM_NEIGHBORHOOD = gd.neighborhood,
    CSM_FIRE_DISTRICT = gd.fireDistrict,
    CSM_SWR_DISTRICT = gd.sewerDistrict,
    CSM_WTR_DISTRICT = gd.waterDistrict,
    CSM_URB_GRWTH_BNDRY = gd.urbanGrowthArea,
    CSM_WILDLAND = gd.wildLand,
      CSM_ZONING1 = gd.zoning1,
      CSM_ZONING2 = gd.zoning2
--    CSM_SCHOOL_DISTRICT = gd.schoolDistrict,  -- do not update fee related
data
--  CSM_TRAFFIC_DISTRICT = gd.trafficDistrict,
--  CSM_PARK_DISTRICT = gd.parkDistrict,
  where CSM_CASENO = caseNo;
  pWrite_Msg( 3, logFileP, 'Exit <<<pUpdate_Case_Extended_Table>>>' );

  EXCEPTION
    WHEN OTHERS THEN
      pWrite_Msg( 3, logFileP, '***Error <<<pUpdate_Case_Extended_Table>>>' );
      pFatal_Error( errFileN, logFileP, 'UpdateC' );

END pUpdate_Case_Extended_Table;


/*
----------------------
pUpdate_Parcel_Table()
----------------------
*/
PROCEDURE pUpdate_Parcel_Table
(
  logFileP IN UTL_FILE.file_type,
  parcelNo PARCEL.PRC_PARCEL_NO%TYPE,
  gd IN rGISDATA
)

IS

BEGIN

  pWrite_Msg( 3, logFileP, 'Enter <<<pUpdate_Parcel_Table>>>' );
/*
  IF gd.quarterNo IS NOT NULL THEN
    UPDATE parcel SET
      PRC_ARCH_PREDICT = gd.archPredict,
      PRC_QUARTER = gd.quarterNo,
        PRC_SECTION = gd.sectionNo,
        PRC_TOWNSHIP = gd.townShipNo,
        PRC_RANGE = gd.rangeNo,
      PRC_JURISDICTION = gd.jurisdiction,
        PRC_ZONING = gd.zoning1,
        PRC_ZONING2 = gd.zoning2,
        PRC_ZONING3 = gd.zoning3,
      PRC_COMP_PLN_DESIGN = gd.compPlanDesign,
        PRC_NEIGHBORHOOD = gd.neighborhood,
      PRC_SCHOOL_DIST = gd.p_schoolDistrict,
      PRC_FIRE_DISTRICT = gd.fireDistrict,
      PRC_TIF_DISTRICT = gd.trafficDistrict,
        PRC_PARK_DISTRICT = gd.parkDistrict,
        PRC_WTR_DISTRICT = gd.waterDistrict,
      PRC_SWR_DISTRICT = gd.sewerDistrict,
      PRC_FLOOD_ZONE = gd.floodZone,
      PRC_CARA = gd.cara,
        PRC_WILDLAND = gd.wildLand,
        PRC_INSPECTOR = gd.inspector,
        PRC_GIS_UPDATED = sysdate
    WHERE PRC_PARCEL_NO = parcelNo;
  ELSE
*/
    -- do not update Quarter Section details
    UPDATE parcel SET
      PRC_ARCH_PREDICT = gd.archPredict,
      PRC_CARA = gd.cara,
      PRC_COMP_PLN_DESIGN = gd.compPlanDesign,
      PRC_CRITICAL_AREA_SF = gd.criticalArea,
      PRC_FIRE_DISTRICT = gd.fireDistrict,
      PRC_FLOOD_ZONE = gd.floodZone,
      PRC_GIS_AREA_SF = gd.gisArea,
        PRC_INSPECTOR = gd.inspector,
--      PRC_JURISDICTION = gd.jurisdiction,
        PRC_NEIGHBORHOOD = gd.neighborhood,
        PRC_PARK_DISTRICT = gd.parkDistrict,
      PRC_SCHOOL_DIST = gd.p_schoolDistrict,
      PRC_SWR_DISTRICT = gd.sewerDistrict,
      PRC_TIF_DISTRICT = gd.trafficDistrict,
      PRC_URB_GRWTH_BNDRY = gd.urbanGrowthArea,
        PRC_WILDLAND = gd.wildLand,
        PRC_WTR_DISTRICT = gd.waterDistrict,
        PRC_ZONING = gd.zoning1,
        PRC_ZONING2 = gd.zoning2,
        PRC_GIS_UPDATED = sysdate
    WHERE PRC_PARCEL_NO = parcelNo;
--  END IF;

  pWrite_Msg( 3, logFileP, 'Exit <<<pUpdate_Parcel_Table>>>' );

  EXCEPTION
    WHEN OTHERS THEN
      pWrite_Msg( 3, logFileP, '***Error <<<pUpdate_Parcel_Table>>>' );
      pFatal_Error( errFileN, logFileP, 'UpdateP' );

END pUpdate_Parcel_Table;


/*
------------
pWrite_Msg()
------------
*/
PROCEDURE pWrite_Msg
(
  logLevel IN NUMBER := 3,
  fp IN UTL_FILE.file_type,
  message IN VARCHAR2
)

IS

BEGIN

  IF logLevel <= MSG_LOG_LEVEL THEN
    UTL_FILE.put_line( fp, message );
  END IF;

END pWrite_Msg;


/*
------------------------
pParse_Quarter_Section()
------------------------
*/
/*
PROCEDURE pParse_Quarter_Section
(
  quarterSection IN VARCHAR2,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 3, logFileP, 'Enter <<<pParse_Quarter_Section>>>' );
  gd.townshipNo := SUBSTR( quarterSection, 1, 1 );
  gd.rangeNo := SUBSTR( quarterSection, 2, 2 );
  gd.sectionNo := SUBSTR( quarterSection, 4, 2 );
  gd.quarterNo := SUBSTR( quarterSection, 6, 1 );
  pWrite_Msg( 3, logFileP, 'Exit <<<pParse_Quarter_Section>>>' );

END pParse_Quarter_Section;
*/

/*
--------------------
fGet_Tidemark_Code()
--------------------
*/
FUNCTION fGet_Tidemark_Code
(
  gItem GIS_TRANSLATOR.ITEM%TYPE,
  gisText GIS_TRANSLATOR.GIS_TEXT%TYPE
)
RETURN VARCHAR2

IS

  code GIS_TRANSLATOR.TM_CODE%TYPE;

BEGIN

  SELECT TM_CODE INTO code
  FROM GIS_TRANSLATOR
  WHERE ITEM = gItem
  AND GIS_TEXT = gisText;

  RETURN code;

  EXCEPTION
    -- if no data found or in case of any errors ...
    WHEN OTHERS THEN
      IF gisText IS NULL THEN
        RETURN BLANK_DATA;
      ELSE
        RETURN NO_TM_CODE;
      END IF;

END fGet_Tidemark_Code;


/*
-----------------
fGet_Total_Tags()
-----------------
*/
FUNCTION fGet_Total_Tags
```

```
(
  domNode XMLDOM.DOMNode,
  tag VARCHAR2
)
RETURN NUMBER

IS

  nodeList xmldom.DOMNodeList;
  curNode xmldom.DOMNode;
  childNode xmldom.DOMNode;
  howMany NUMBER := 0;
  noOfNotNullValues NUMBER := 0;

BEGIN

  nodeList := xslprocessor.selectNodes( domNode, tag );
  howMany := xmldom.getLength( nodeList );

  FOR i IN 0..howMany-1
  LOOP
    curNode := XMLDOM.item( nodeList, i );
    childNode := XMLDOM.getFirstChild( curNode );
    IF XMLDOM.IsNull( childNode ) = FALSE THEN
        noOfNotNullValues := noOfNotNullValues +1;
    END IF;
  END LOOP;

  RETURN noOfNotNullValues;

END fGet_Total_Tags;


/*
----------------
fGet_Tag_Value()
----------------
*/
FUNCTION fGet_Tag_Value
(
  domNode XMLDOM.DOMNode,
  tag VARCHAR2,
--  howLong NUMBER,
  nTag NUMBER := 1
)
RETURN VARCHAR2

IS

  nodeList xmldom.DOMNodeList;
  curNode xmldom.DOMNode;
  childNode xmldom.DOMNode;
  howMany NUMBER := 0;
  noOfNotNullValues NUMBER := 0;

BEGIN

  nodeList := xslprocessor.selectNodes( domNode, tag );
  howMany := xmldom.getLength( nodeList );

  -- find the first non-empty value and return
  FOR i IN 0..howMany-1
  LOOP
    curNode := XMLDOM.item( nodeList, i );
    childNode := XMLDOM.getFirstChild( curNode );
    IF XMLDOM.IsNull( childNode ) = FALSE THEN
        noOfNotNullValues := noOfNotNullValues +1;
        IF noOfNotNullValues = nTag THEN
          RETURN Trim( XMLDOM.getNodeValue( childNode ) );
--        pWrite_Msg( 3, logFileP, '<' || tagValue );
        END IF;
    END IF;
  END LOOP;

  RETURN NULL;

END fGet_Tag_Value;

/*
--------------
pFatal_Error()
--------------
*/
PROCEDURE pFatal_Error
(
  errFileN IN VARCHAR2,
  logFileP IN UTL_FILE.file_type,
  errType GIS_ERR_MSG.ERR_TYPE%TYPE
)

IS

  errNo NUMBER := 0;
  errMsg VARCHAR2( 100 ) := NULL;
  errFileP UTL_FILE.file_type;

BEGIN

  pWrite_Msg( 3, logFileP, 'Enter <<<pFatal_Error>>>' );
  errFileP := utl_file.fopen( DirN, errFileN, 'a' );

  SELECT ERR_NO, ERR_MSG
  INTO errNo, errMsg
  FROM GIS_ERR_MSG
  WHERE ERR_TYPE = errType;

  pWrite_Msg( 1, logFileP, errNo || ': ' || errMsg );
  pWrite_Msg( 3, logFileP, 'Exit <<<pFatal_Error>>>' );

--  RAISE_APPLICATION_ERROR( errNo, '[GISlink] ' || errMsg || '***' );

  EXCEPTION

    WHEN NO_DATA_FOUND THEN
      pWrite_End( logFileP );
        UTL_FILE.fclose_all;
        RAISE_APPLICATION_ERROR
        (
          -20901, '[GISlink] UnDefined Error Condition***'
        );

/*
```

```
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR
        (
          -20903, '[GISlink] Unknown Error Condition***'
        );
        UTL_FILE.fclose_all;
*/
END pFatal_Error;


/*
------------------
fIs_Valid_Parcel()
------------------
*/
FUNCTION fIs_Valid_Parcel
(
  parcelNo PARCEL.PRC_PARCEL_NO%TYPE
)

RETURN BOOLEAN

IS

  p_id NUMBER;

BEGIN

  IF parcelNo = '0' OR parcelNo = '9999999999' THEN
    RETURN FALSE;
  END IF;

  p_id := to_number( parcelNo );
  RETURN TRUE;

  EXCEPTION
    WHEN OTHERS THEN
        RETURN FALSE;

END fIs_Valid_Parcel;


/*
----------------------
pWrite_Parcel_Number()
----------------------
*/
PROCEDURE pWrite_Parcel_Number
(
  caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
  parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE
)

IS

  logFileP UTL_FILE.file_type;
  tmpFileP UTL_FILE.file_type;
  DB_INSTANCE VARCHAR2( 10 );

BEGIN
```

```
  SELECT lower( NAME ) INTO DB_INSTANCE FROM SYS.v_$database;
  DirN := '/v01/' || DB_INSTANCE || '/dump/gislog';
  tmpFileN := caseNo || TMP_FILE_SUFFIX;
  tmpFileP := utl_file.fopen( DirN, tmpFileN, 'w' );
  UTL_FILE.put_line( tmpFileP, parcelNo );
  UTL_FILE.fclose( tmpFileP );

  EXCEPTION
    WHEN OTHERS THEN
      logFileN := caseNo || LOG_FILE_SUFFIX;
      logFileP := utl_file.fopen( DirN, logFileN, 'a' );
      pFatal_Error( errFileN, logFileP, 'pWrite_Parcel_Number_1' );

END pWrite_Parcel_Number;


/*
---------------------
fRead_Parcel_Number()
---------------------
*/
FUNCTION fRead_Parcel_Number
(
  caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
  parcelNo OUT PARCEL.PRC_PARCEL_NO%TYPE
)

RETURN BOOLEAN

IS

  tmpFileP UTL_FILE.file_type;
  eDummyParcel EXCEPTION;

BEGIN

  tmpFileN := caseNo || TMP_FILE_SUFFIX;
  tmpFileP := utl_file.fopen( DirN, tmpFileN, 'r' );
  UTL_FILE.get_line( tmpFileP, parcelNo );
  UTL_FILE.fclose( tmpFileP );

  RETURN fIs_Valid_Parcel( parcelNo );

  EXCEPTION
    WHEN OTHERS THEN
      RETURN FALSE;

END fRead_Parcel_Number;


/*
-------------------------
fCopy_Local_Parcel_Data()
-------------------------
*/
FUNCTION fCopy_Local_Parcel_Data
(
  logFileP IN UTL_FILE.file_type,
  caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
  parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
  gd OUT rGISDATA
)

RETURN BOOLEAN

IS

  last_gis_updated_date DATE;

BEGIN

  pWrite_Msg( 3, logFileP, 'Enter <<<fCopy_Local_Parcel_Data>>>' );

  SELECT PRC_GIS_UPDATED INTO last_gis_updated_date
  FROM PARCEL WHERE PRC_PARCEL_NO = parcelNo;

  IF last_gis_updated_date IS NULL THEN
    pWrite_Msg( 2, logFileP, 'Parcel No ' ||
                 parcelNo || ' was never updated by GIS!'
               );
    pWrite_Msg( 3, logFileP, 'Exit <<<fCopy_Local_Parcel_Data>>>' );
    RETURN FALSE;
  END IF;

  pWrite_Msg( 2, logFileP, 'Parcel No ' || parcelNo ||
               ' was last updated by GIS on: ' ||
               to_char( last_gis_updated_date, 'MM/DD/YYYY HH:MI:SS AM' )
             );

  pWrite_Msg( 2, logFileP, to_char( (sysdate - last_gis_updated_date),
'999999.99' ) ||
                         ' days ago'
            );

  IF sysdate - last_gis_updated_date < ACCEPTABLE_DAYS THEN
    SELECT
      PRC_ARCH_PREDICT,
      PRC_CARA,
      PRC_COMP_PLN_DESIGN,
      PRC_CRITICAL_AREA_SF,
      PRC_FLOOD_ZONE,
      PRC_GIS_AREA_SF,
        PRC_INSPECTOR,
      PRC_JURISDICTION,
        PRC_NEIGHBORHOOD,
      PRC_FIRE_DISTRICT,
      PRC_PARK_DISTRICT,
      PRC_SCHOOL_DIST,
      PRC_SWR_DISTRICT,
      PRC_TIF_DISTRICT,
      PRC_WTR_DISTRICT,
      PRC_URB_GRWTH_BNDRY,
        PRC_WILDLAND,
        PRC_ZONING,
        PRC_ZONING2
    INTO
      gd.archPredict,
      gd.cara,
      gd.compPlanDesign,
      gd.criticalArea,
      gd.floodZone,
      gd.gisArea,
      gd.inspector,
      gd.jurisdiction,
      gd.neighborhood,
      gd.fireDistrict,
      gd.parkDistrict,
      gd.p_schoolDistrict,
      gd.sewerDistrict,
      gd.trafficDistrict,
      gd.waterDistrict,
      gd.urbanGrowthArea,
      gd.wildLand,
      gd.zoning1,
      gd.zoning2
    FROM PARCEL
    WHERE PRC_PARCEL_NO = parcelNo;

    gd.schoolDistrict := fGet_Tidemark_Code( 'SCHOOL', to_char(
gd.p_schoolDistrict ) );

    IF gd.neighborhood = MULTIPLE_DATA THEN
      gd.m_neighborhood := 'Y';
    ELSE
      gd.m_neighborhood := 'N';
    END IF;

    pWrite_Msg( 3, logFileP, 'Exit <<<fCopy_Local_Parcel_Data>>>' );
    RETURN TRUE;

  END IF;

  pWrite_Msg( 3, logFileP, 'Exit <<<fCopy_Local_Parcel_Data>>>' );
  RETURN FALSE;

  EXCEPTION
    WHEN OTHERS THEN
      pWrite_Msg( 3, logFileP, '***Error <<<fCopy_Local_Parcel_Data>>>' );
      RETURN FALSE;

END fCopy_Local_Parcel_Data;


/*
--------------------
fCopy_Arch_Predict()
--------------------
*/
PROCEDURE fCopy_Arch_Predict
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Arch Probability]' );
  howMany := fGet_Total_Tags( domNode, ARCH_PREDICT_TAG );

  IF howMany <= 1 THEN
```

```
    gisText := fGet_Tag_Value( domNode, ARCH_PREDICT_TAG );
    pWrite_Msg( 3, logFileP, ' [GIS]: ' || gisText );
    gd.archPredict := fGet_Tidemark_Code( 'APREDICT', gisText );
  ELSE
    gd.archPredict := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.archPredict );

END fCopy_Arch_Predict;

/*
PROCEDURE fCopy_Quarter_Section
(
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

  quarterSection VARCHAR2( 6 ) := NULL;

BEGIN

  pWrite_Msg( 1, logFileP, '[QuarterSection]' );
  howMany := fGet_Total_Tags( domNode, QUARTER_SECTION_TAG );

  IF howMany = 1 THEN
    quarterSection := fGet_Tag_Value( domNode, QUARTER_SECTION_TAG );
    pWrite_Msg( 3, logFileP, ' [GIS]: ' || quarterSection );
    pParse_Quarter_Section( quarterSection, gd );
    pWrite_Msg( 1, logFileP, ' [TM] TownShipNo: ' || gd.townShipNo );
    pWrite_Msg( 1, logFileP, ' [TM] RangeNo: ' || gd.rangeNo );
    pWrite_Msg( 1, logFileP, ' [TM] SectionNo: ' || gd.sectionNo );
    pWrite_Msg( 1, logFileP, ' [TM] QuarterNo: ' || gd.quarterNo );
  ELSE
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

END fCopy_Quarter_Section;
*/

/*
------------------------
fCopy_Gis_Jurisdiction()
------------------------
*/
PROCEDURE fCopy_Gis_Jurisdiction
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[GIS Jurisdiction]' );
  howMany := fGet_Total_Tags( domNode, JURISDICTION_TAG );
  IF howMany <= 1 THEN
    gisText := fGet_Tag_Value( domNode, JURISDICTION_TAG );
    pWrite_Msg( 1, logFileP, ' [GIS]: ' || gisText );
    gd.jurisdiction := fGet_Tidemark_Code( 'JURISDICTION', gisText );
  ELSE
    gd.jurisdiction := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.jurisdiction );

END fCopy_Gis_Jurisdiction;


/*
---------------------------
fCopy_Parcel_Jurisdiction()
---------------------------
*/
PROCEDURE fCopy_Parcel_Jurisdiction
(
  logFileP IN UTL_FILE.file_type,
  parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Parcel Jurisdiction]' );

  SELECT PRC_JURISDICTION INTO gd.jurisdiction
  FROM PARCEL
  WHERE PRC_PARCEL_NO = parcelNo;

END fCopy_Parcel_Jurisdiction;


/*
--------------
fCopy_Zoning()
--------------
*/
PROCEDURE fCopy_Zoning
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Zoning]' );
  howMany := fGet_Total_Tags( domNode, ZONING_TAG );

  IF howMany <= 1 THEN
    gd.zoning1 :=
      NVL( fGet_Tag_Value( domNode, ZONING_TAG, 1 ), BLANK_DATA );
    pWrite_Msg( 1, logFileP, ' [TM] Zone1: ' || gd.zoning1 );

  ELSIF howMany = 2 THEN
    -- copy both zone1 and zone2
    gd.zoning1 :=
      NVL( fGet_Tag_Value( domNode, ZONING_TAG, 1 ), BLANK_DATA );
    pWrite_Msg( 1, logFileP, ' [TM] Zone1: ' || gd.zoning1 );
    gd.zoning2 :=
      NVL( fGet_Tag_Value( domNode, ZONING_TAG, 2 ), BLANK_DATA );
    pWrite_Msg( 1, logFileP, ' [TM] Zone2: ' || gd.zoning2 );
/*
  ELSIF howMany = 3 THEN
    -- copy zone1, zone2 and zone3
    gd.zoning1 :=
      NVL( fGet_Tag_Value( domNode, ZONING_TAG, 1 ), BLANK_DATA );
    pWrite_Msg( 1, logFileP, ' [TM] Zone1: ' || gd.zoning1 );
    gd.zoning2 :=
      NVL( fGet_Tag_Value( domNode, ZONING_TAG, 2 ), BLANK_DATA );
    pWrite_Msg( 1, logFileP, ' [TM] Zone2: ' || gd.zoning2 );
    gd.zoning3 :=
      NVL( fGet_Tag_Value( domNode, ZONING_TAG, 3 ), BLANK_DATA );
    pWrite_Msg( 1, logFileP, ' [TM] Zone3: ' || gd.zoning3 );
*/
  ELSE
    -- too many zones; so populate with '*'s
    gd.zoning1 := MULTIPLE_DATA;
--    gd.zoning2 := MULTIPLE_DATA;
--    gd.zoning3 := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  Exception
    WHEN OTHERS THEN
      pWrite_Msg( 3, logFileP, '***Error <<<fCopy_Zoning>>>' );

END fCopy_Zoning;


/*
----------------------
fCopy_Complan_Design()
----------------------
*/
PROCEDURE fCopy_Complan_Design
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Complan Design]' );
  howMany := fGet_Total_Tags( domNode, COMPLAN_DESIGN_TAG );

  IF howMany <= 1 THEN
--    gisText := fGet_Tag_Value( domNode, COMPLAN_DESIGN_TAG );
--    pWrite_Msg( 3, logFileP, ' [GIS]: ' || gisText );
--    gd.compPlanDesign := fGet_Tidemark_Code( 'COMPLAN', gisText );
    gd.compPlanDesign :=
      NVL( fGet_Tag_Value( domNode, COMPLAN_DESIGN_TAG ), BLANK_DATA );
  ELSE
    gd.compPlanDesign := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.compPlanDesign );

END fCopy_Complan_Design;


/*
---------------------
fCopy_Critical_Area()
---------------------
*/
PROCEDURE fCopy_Critical_Area
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[CRITICAL AREA]' );
  howMany := fGet_Total_Tags( domNode, CRITICAL_AREA_TAG );
  If howMany > 1 Then
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  End If;

  gisText := fGet_Tag_Value( domNode, CRITICAL_AREA_TAG );
  pWrite_Msg( 1, logFileP, ' [GIS]: ' || gisText );
  -- Jurisdiction and UGA share the Tidemark code and picklist (for now)
  gd.criticalArea := To_Number( NVL( gisText, 0 ) );

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.criticalArea );

END fCopy_Critical_Area;


/*
----------------
fCopy_Gis_Sqft()
----------------
*/
PROCEDURE fCopy_Gis_Sqft
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN
```

```
  pWrite_Msg( 1, logFileP, '[GIS SQFT]' );

  howMany := fGet_Total_Tags( domNode, ASSR_SQFT_TAG );
  If howMany > 1 Then
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  End If;

  gisText := fGet_Tag_Value( domNode, ASSR_SQFT_TAG );
  pWrite_Msg( 1, logFileP, ' [ASR]: ' || gisText );


  If To_Number( NVL( gisText, 0 ) ) = 0 Then
    howMany := fGet_Total_Tags( domNode, GIS_SQFT_TAG );
    If howMany > 1 Then
      pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
    End If;

    gisText := fGet_Tag_Value( domNode, GIS_SQFT_TAG );
    pWrite_Msg( 1, logFileP, ' [GIS]: ' || gisText );
  End If;

  gd.gisArea := To_Number( NVL( gisText, 0 ) );
  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.gisArea );

END fCopy_Gis_Sqft;


/*
--------------------
fCopy_Neighborhood()
--------------------
*/
PROCEDURE fCopy_Neighborhood
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Neighborhood]' );
  gd.m_neighborhood := 'N';
  howMany := fGet_Total_Tags( domNode, NEIGHBORHOOD_TAG );
  IF howMany <= 1 THEN
    gd.neighborhood :=
      NVL( fGet_Tag_Value( domNode, NEIGHBORHOOD_TAG ), BLANK_DATA );
  ELSE
    gd.neighborhood := MULTIPLE_DATA;
      gd.m_neighborhood := 'Y';
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.neighborhood );
  pWrite_Msg( 1, logFileP, ' Multiple?: ' || gd.m_neighborhood );

END fCopy_Neighborhood;
```

```
/*
-----------------------
fCopy_School_District()
-----------------------
*/
PROCEDURE fCopy_School_District
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[School District]' );
  howMany := fGet_Total_Tags( domNode, SCHOOL_DISTRICT_TAG );

  IF howMany <= 1 THEN
    gisText := fGet_Tag_Value( domNode, SCHOOL_DISTRICT_TAG );
    pWrite_Msg( 3, logFileP, ' [GIS]: ' || gisText );
    gd.schoolDistrict := fGet_Tidemark_Code( 'SCHOOL', gisText );
    -- Convert to number to store in Parcel Table
    gd.p_schoolDistrict := to_number( gisText );
  ELSE
    gd.schoolDistrict := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM] Parcel: ' || gd.p_schoolDistrict );
  pWrite_Msg( 1, logFileP, ' [TM] Case: ' || gd.schoolDistrict );

END fCopy_School_District;


/*
---------------------
fCopy_Fire_District()
---------------------
*/
PROCEDURE fCopy_Fire_District
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Fire District]' );
  howMany := fGet_Total_Tags( domNode, FIRE_DISTRICT_TAG );

  IF howMany <= 1 THEN
    gisText := fGet_Tag_Value( domNode, FIRE_DISTRICT_TAG );
    gd.fireDistrict := fGet_Tag_Value( domNode, FIRE_DISTRICT_TAG );
  ELSE
    gd.fireDistrict := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.fireDistrict );

END fCopy_Fire_District;


/*
------------------------
fCopy_Traffic_District()
------------------------
*/
PROCEDURE fCopy_Traffic_District
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Traffic District]' );
  howMany := fGet_Total_Tags( domNode, TRAFFIC_DISTRICT_TAG );

  IF howMany <= 1 THEN
    gisText := fGet_Tag_Value( domNode, TRAFFIC_DISTRICT_TAG );
    pWrite_Msg( 3, logFileP, ' [GIS]: ' || gisText );
    gd.trafficDistrict := fGet_Tidemark_Code( 'TRAFFIC', gisText );
  ELSE
    gd.trafficDistrict := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.trafficDistrict );

END fCopy_Traffic_District;


/*
---------------------
fCopy_Park_District()
---------------------
*/
PROCEDURE fCopy_Park_District
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Park District]' );
  howMany := fGet_Total_Tags( domNode, PARK_DISTRICT_TAG );

  IF howMany <= 1 THEN
    gd.parkDistrict :=
      NVL( fGet_Tag_Value( domNode, PARK_DISTRICT_TAG ), BLANK_DATA );
  ELSE
    gd.parkDistrict := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.parkDistrict );

END fCopy_Park_District;


/*
----------------------
fCopy_Sewer_District()
----------------------
*/
PROCEDURE fCopy_Sewer_District
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Sewer District]' );
  howMany := fGet_Total_Tags( domNode, SEWER_DISTRICT_TAG );

  IF howMany <= 1 THEN
    gisText := fGet_Tag_Value( domNode, SEWER_DISTRICT_TAG );
    pWrite_Msg( 3, logFileP, ' [GIS]: ' || gisText );
    gd.sewerDistrict := fGet_Tidemark_Code( 'SEWER', gisText );
  ELSE
    gd.sewerDistrict := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.sewerDistrict );

END fCopy_Sewer_District;


/*
----------------------
fCopy_Water_District()
----------------------
*/
PROCEDURE fCopy_Water_District
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Water District]' );
  howMany := fGet_Total_Tags( domNode, WATER_DISTRICT_TAG );

  IF howMany <= 1 THEN
    gisText := fGet_Tag_Value( domNode, WATER_DISTRICT_TAG );
    pWrite_Msg( 3, logFileP, ' [GIS]: ' || gisText );
    gd.waterDistrict := fGet_Tidemark_Code( 'WATER', gisText );
  ELSE
    gd.waterDistrict := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.waterDistrict );

END fCopy_Water_District;


/*
----------------------
fCopy_Inspector_Area()
----------------------
*/
PROCEDURE fCopy_Inspector_Area
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[Inspector Area]' );
  howMany := fGet_Total_Tags( domNode, INSPECTOR_TAG );

  IF howMany <= 1 THEN
    gd.inspector :=
      NVL( fGet_Tag_Value( domNode, INSPECTOR_TAG ), BLANK_DATA );
  ELSE
    gd.inspector := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.inspector );

END fCopy_Inspector_Area;


/*
------------
fCopy_Cara()
------------
*/
PROCEDURE fCopy_Cara
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS
```

```
BEGIN

  pWrite_Msg( 1, logFileP, '[CARA]' );
  howMany := fGet_Total_Tags( domNode, CARA_TAG );

  IF howMany <= 1 THEN
    gd.cara :=
      NVL( fGet_Tag_Value( domNode, CARA_TAG ), BLANK_DATA );
  ELSE
    gd.cara := MULTIPLE_DATA;
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.cara );

END fCopy_Cara;


/*
------------------
fCopy_Flood_Zone()
------------------
*/
PROCEDURE fCopy_Flood_Zone
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[FLOOD_ZONE]' );
  howMany := fGet_Total_Tags( domNode, FLOOD_ZONE_TAG );

  IF howMany <= 1 THEN
    gd.floodZone :=
      NVL( fGet_Tag_Value( domNode, FLOOD_ZONE_TAG ), BLANK_DATA );
  ELSE
    gd.floodZone := '*';
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.floodZone );

END fCopy_Flood_Zone;


/*
----------------
fCopy_Wildland()
----------------
*/
PROCEDURE fCopy_Wildland
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
```

```
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[WILDLAND]' );
  howMany := fGet_Total_Tags( domNode, WILDLAND_TAG );

  IF howMany <= 1 THEN
    gd.wildLand :=
      NVL( fGet_Tag_Value( domNode, WILDLAND_TAG ), BLANK_DATA );
  ELSE
    gd.wildLand := '*';
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  END IF;

  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.wildland );

END fCopy_Wildland;


/*
-------------------------
fCopy_Urban_Growth_Area()
-------------------------
*/
PROCEDURE fCopy_Urban_Growth_Area
(
  logFileP IN UTL_FILE.file_type,
  domNode IN XMLDOM.DOMNode,
  gd IN OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 1, logFileP, '[URBAN GROWTH AREA]' );
  howMany := fGet_Total_Tags( domNode, URBAN_GROWTH_AREA_TAG );
  If howMany > 1 Then
    pWrite_Msg( 3, logFileP, ' has ' || howMany || ' values in GIS!' );
  End If;

  gisText := fGet_Tag_Value( domNode, URBAN_GROWTH_AREA_TAG );
  pWrite_Msg( 1, logFileP, ' [GIS]: ' || gisText );
  -- Jurisdiction and UGA share the Tidemark code and picklist (for now)
  gd.urbanGrowthArea := fGet_Tidemark_Code( 'JURISDICTION', gisText );
  pWrite_Msg( 1, logFileP, ' [TM]: ' || gd.urbanGrowthArea );

END fCopy_Urban_Growth_Area;


/*
------------------------
pCheck_For_Parcel_Tags()
------------------------
*/
PROCEDURE pCheck_For_Parcel_Tags
(
  logFileP IN UTL_FILE.file_type,
  parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
  domNode IN XMLDOM.DOMNode
)

IS

BEGIN

  pWrite_Msg( 3, logFileP, 'Enter <<<pCheck_For_Parcel_Tags>>>' );

  -- Columbia River Gorge NSA
  pWrite_Msg( 3, logFileP, ' [CR Gorge Parcel Tag]' );
  gisText := fGet_Tag_Value( domNode, CR_GORGE_TAG );
  pWrite_Msg( 1, logFileP, ' [GIS]: ' || gisText );

  IF gisText = 'Yes' THEN
    pAdd_Parcel_Tag( logFileP, parcelNo, CRG_ATTR_CODE, CRG_ATTR_ORDER );
  END IF;

  -- Salmon Creek Moratorium
--  pWrite_Msg( 3, logFileP, ' [Salmon Creek Parcel Tag]' );

  -- Rural Land Division Moratorium
  pWrite_Msg( 3, logFileP, ' [Rural Land Division Parcel Tag]' );
  gisText := fGet_Tag_Value( domNode, SCM_AREA_TAG );
  pWrite_Msg( 1, logFileP, ' [GIS]: ' || gisText );

/*
  IF gisText = 'Salmon Creek' THEN
    pAdd_Parcel_Tag( parcelNo, SCM_ATTR_CODE, SCM_ATTR_ORDER );
  END IF;
*/

  IF gisText = 'Rural Land Division Moratorium' THEN
    pAdd_Parcel_Tag( logFileP, parcelNo, RLDM_ATTR_CODE, RLDM_ATTR_ORDER );
  END IF;

  pWrite_Msg( 3, logFileP, 'Exit <<<pCheck_For_Parcel_Tags>>>' );

END pCheck_For_Parcel_Tags;


/*
-----------------
pAdd_Parcel_Tag()
-----------------
*/
PROCEDURE pAdd_Parcel_Tag
(
  logFileP IN UTL_FILE.file_type,
  parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
  attribCode IN ATTRIB_LIB.ATTR_CODE%TYPE,
  attribOrder IN ATTRIB_LIB.ATTR_ORDERING%TYPE
)

IS

  c NUMBER := 0;
  tagWarning ATTRIB_LIB.ATTR_WARNING%TYPE;
  tagDescription ATTRIB_LIB.ATTR_DESCRIPTION%TYPE;
```

```
BEGIN

  pWrite_Msg( 3, logFileP, 'Enter <<<pAdd_Parcel_Tag>>>' );

  SELECT COUNT(*) INTO c
  FROM PARCEL_ATTRIB
  WHERE PRC_PARCEL_NO = parcelNo
  AND ATTR_CODE = attribCode;

  IF c = 0 THEN
    SELECT ATTR_WARNING, ATTR_DESCRIPTION
    INTO tagWarning, tagDescription
    FROM ATTRIB_LIB
    WHERE ATTR_CODE = attribCode
    AND ATTR_ORDERING = attribOrder;

    INSERT INTO PARCEL_ATTRIB
    (
      PRCLATTR_ID,
      PRC_PARCEL_NO,
      PRC_AVP_NO,
      ATTR_CODE,
      ATTR_USE_VALUE,
      ATTR_VALUE,
      ATTR_WARNING,
      ATTR_DESCRIPTION,
      PRCLATTR_UPDATEBY,
      PRCLATTR_UPDATED
    )
    VALUES
    (
      to_char( sysdate, 'YYYYMMDDHH24MiSSSSS' ) || '0',
      parcelNo,
      0,
      attribCode,
      'N',
      NULL,
      tagWarning,
      tagDescription,
      'GIS',
      sysdate
    );
    pWrite_Msg( 1, logFileP, ' Parcel tag added' );
  ELSE
    pWrite_Msg( 1, logFileP, ' Parcel tag exists! Not added.' );
  END IF;

  pWrite_Msg( 3, logFileP, 'Exit <<<pAdd_Parcel_Tag>>>' );

END pAdd_Parcel_Tag;


/*
-----------------
pInsert_Gis_Log()
-----------------
*/
PROCEDURE pInsert_Gis_Log
(
  logFileP IN UTL_FILE.file_type,
  caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
  parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
  action IN GIS_LOG.GIS_ACTION%TYPE,
  status IN GIS_LOG.GIS_STATUS%TYPE,
  comments IN GIS_LOG.GIS_COMMENTS%TYPE := NULL
)

IS

  userName PP_USER.USER_NAME%TYPE;

BEGIN

  pWrite_Msg( 3, logFileP, 'Enter <<<pInsert_Gis_Log>>>' );

  SELECT user_name INTO userName FROM PP_USER
  WHERE user_id = ( SELECT lower(user) FROM DUAL );

  INSERT INTO GIS_LOG
  (
    GIS_LOG_ID,
    CSM_CASENO,
    PRC_PARCEL_NO,
    GIS_ACTION,
    GIS_STATUS,
    GIS_COMMENTS,
    WIN_SERVER,
    MLQ_DURATION,
    GIS_DURATION,
    GIS_UPDATED,
    GIS_UPDATEDBY
  )
  VALUES
  (
    to_char( sysdate, 'YYYYMMDDHH24MiSSSSS' ) || '0',
    caseNo,
    parcelNo,
    action,
    status,
    comments,
    winServer,
    ( MLQ_END - MLQ_START ) * 86400,
    ( SYSDATE - GIS_START ) * 86400,
    SYSDATE,
    userName
  );

  pWrite_Msg( 3, logFileP, 'Exit <<<pInsert_Gis_Log>>>' );

END pInsert_Gis_Log;


/*
--------------
pParse_Error()
--------------
*/
PROCEDURE pParse_Error
(
  logFileP IN UTL_FILE.file_type,
  eType IN GIS_ERR_MSG.ERR_TYPE%TYPE,
  caseNo IN CASE_EXTENDED.CSM_CASENO%TYPE,
  parcelNo IN PARCEL.PRC_PARCEL_NO%TYPE,
  action IN GIS_LOG.GIS_ACTION%TYPE,
  gd OUT rGISDATA
)

IS

BEGIN

  pWrite_Msg( 0, logFileP, eType );
  pWrite_Msg( 0, logFileP, 'Enter <<<pParse_Error>>>' );

  IF fCopy_Local_Parcel_Data( logFileP, caseNo, parcelNo, gd ) = TRUE THEN
    pWrite_Msg( 0, logFileP, 'Successfully copied GIS data from Parcel!' );
    pInsert_Gis_Log
    (
      logFileP, caseNo, parcelNo, action, 'Failure', 'Copied Local Data'
    );
  ELSE
    pWrite_Msg( 0, logFileP, 'No/Old GIS data in Parcel' );
    pInsert_Gis_Log
    (
      logFileP, caseNo, parcelNo, action, 'Failure', 'No/Old Local Data'
    );
  END IF;

  pFatal_Error( errFileN, logFileP, eType );
  pWrite_Msg( 0, logFileP, 'Exit <<<pParse_Error>>>' );

END pParse_Error;


/*
--------------
pWrite_Start()
--------------
*/
PROCEDURE pWrite_Start
(
  logFileP IN UTL_FILE.file_type
)

IS

BEGIN

  winServer := 'none';
  GIS_START := SYSDATE;
  MLQ_START := GIS_START;
  MLQ_END := GIS_START;
  pWrite_Msg( 0, logFileP, '+-----------------------------------------+Start'
);
  pWrite_Msg( 0, logFileP, 'Date/Time: ' || TO_CHAR( SYSDATE, 'MM/DD/YYYY
HH:MI:SS AM' ));

END pWrite_Start;


/*
------------
pWrite_End()
```

```
------------
*/
PROCEDURE pWrite_End
(
  logFileP IN UTL_FILE.file_type
)

IS

BEGIN

  pWrite_Msg( 0, logFileP, '+-----------------------------------------+End' );

END pWrite_END;


END GIS;
/
```