**RD - Desenvolvedor Business Intelligence - Projeto Desafio**

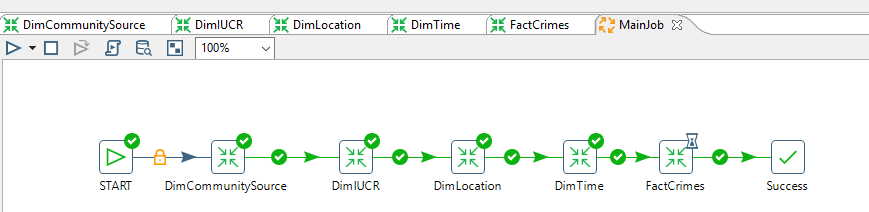
Guilherme Ocker Ribeiro

**ETL e definição do modelo ER:**

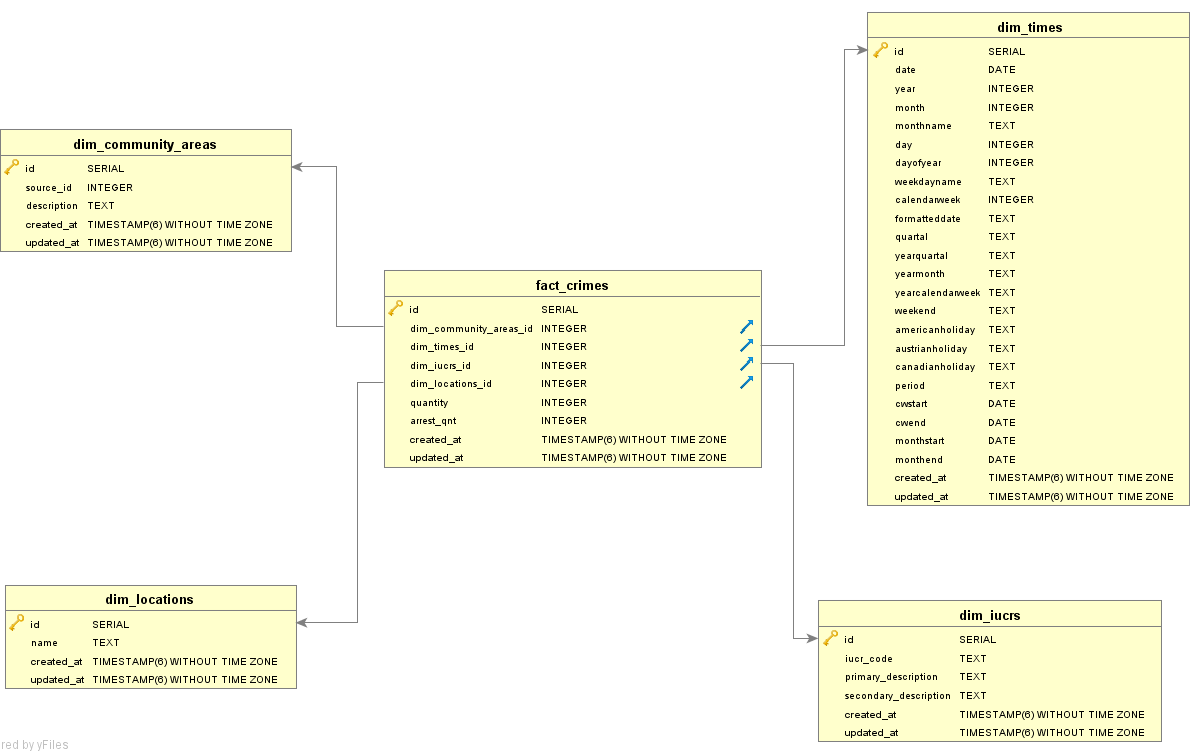
Foi utilizada a ferramenta PDI - Kettle, da suite Pentaho. Ferramenta bem reconhecida no mercado, agnóstica quanto a fontes e destinos e se trata de uma ferramenta onde possuo experiência de alguns anos. ([Download](http://community.pentaho.com/projects/data-integration/))

Para facilitar o início do desenvolvimento do processo de ETL, foi extraído uma amostra da carga completa, com algumas milhares de linhas, facilitando o processo de testes iniciais, construção do modelo e debug, além da análise completa do arquivo através de uma exportação do arquivo “cru” para a base de dados, possibilitando visualizar a natureza dos dados e seus relacionamentos.

**MainJob:**



**Modelagem Dimensional:**



A tabela *Community Areas* foi extraída a partir deste [link](https://data.cityofchicago.org/Facilities-Geographic-Boundaries/Boundaries-Community-Areas-current-/cauq-8yn6) e importado para dentro do modelo a partir de um ETL específico. A tabela IUCR (*Illinois Uniform Crime Reporting*) foi extraída deste [link](https://data.cityofchicago.org/Public-Safety/Chicago-Police-Department-Illinois-Uniform-Crime-R/c7ck-438e) e também tratada e importada para dentro do modelo através de ETL específico.

A Dimensão de tempo *dim\_times* foi construída através de um script dinâmico encontrado neste [link](https://wiki.postgresql.org/wiki/Date_and_Time_dimensions), e a partir dele, montado o arquivo .csv usado no ETL.

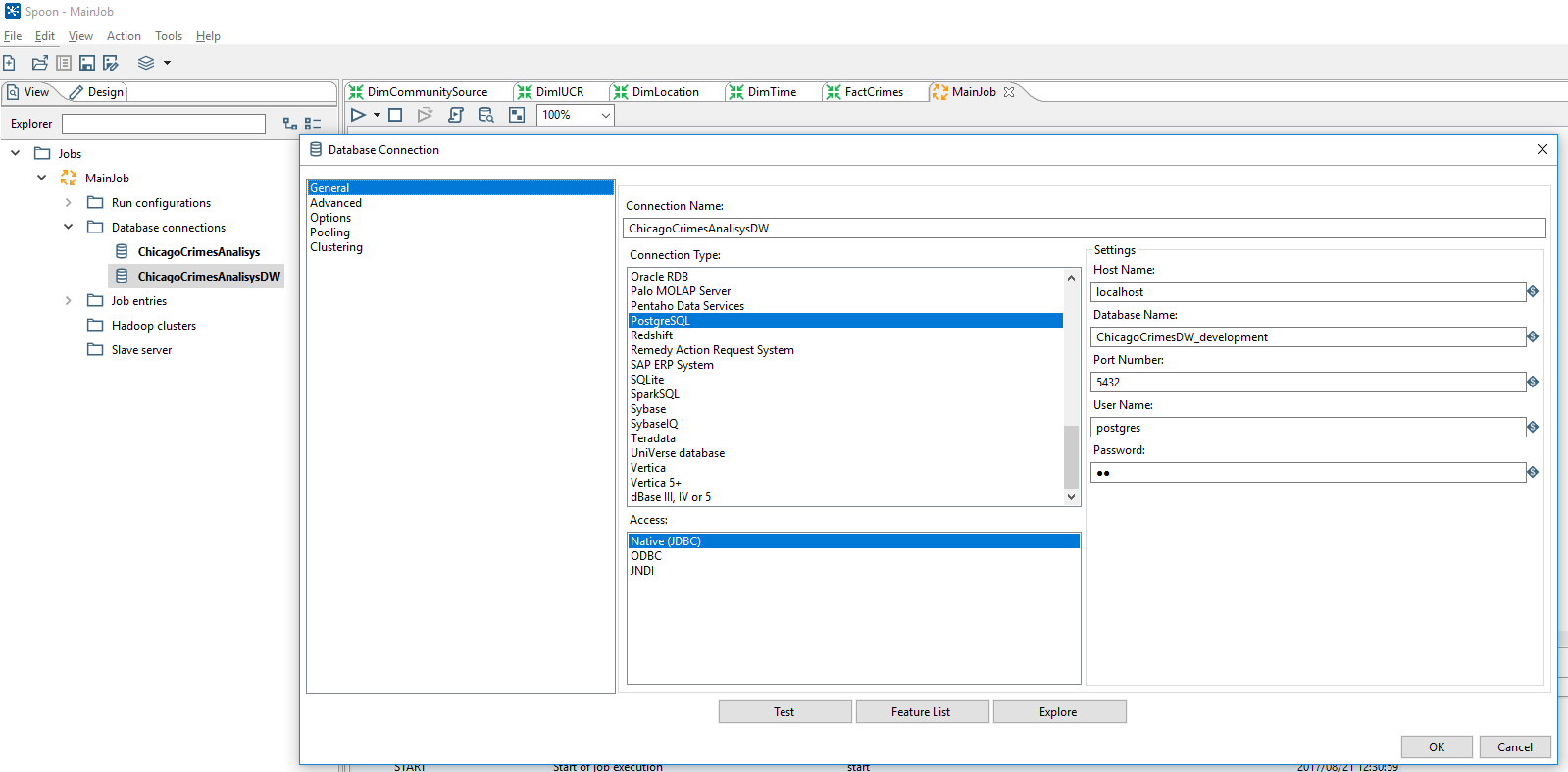
**Configurações e Execução do ETL**

[Download](http://community.pentaho.com/projects/data-integration/) - Executar ~\pdi-ce-7.1.0.0-12\data-integration\**Spoon.bat**

Se não abrir, pode ser necessário alterar as configurações de criação da JVM, através dos seguintes passos:

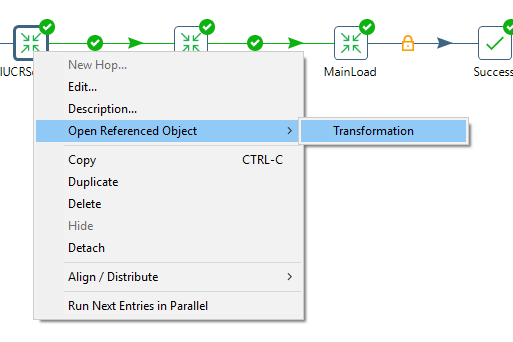
* Editar o arquivo Spoon.bat, próximo a linha 94, identificar e alterar para os seguintes parâmetros: ***PENTAHO\_DI\_JAVA\_OPTIONS="-Xms512m" "-Xmx1024m" "-XX:MaxPermSize=256m"***
* Executar novamente o arquivo Spoon.bat.

**Configuração de banco:**



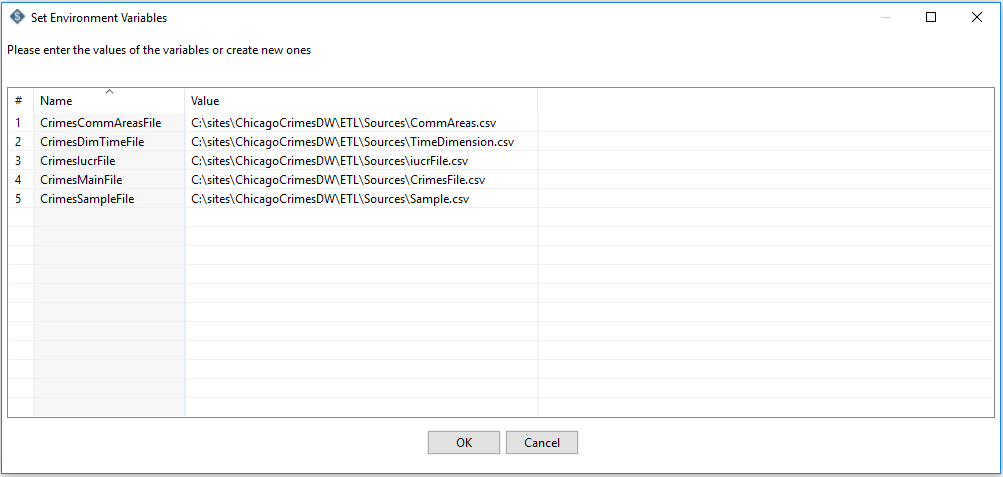
**Arquivo de Job:**

Abrir o arquivo de *job ChicagoCrimes.kjb* dentro da pasta de ETL do projeto. A partir dele, é possível “navegar” até os outros arquivos *transformation* se for de interesse:



**Variaveis de ambiente Kettle:**

Serão utilizadas variáveis de ambiente para facilitar a identificação dos arquivos de fonte. Configurar de acordo com o print e valores da tabela abaixo.



O arquivo *CrimesFile.csv* deve ser baixado através deste [link](https://data.cityofchicago.org/api/views/ijzp-q8t2/rows.csv?accessType=DOWNLOAD), renomeado e colocado na pasta Sources, como mostrado abaixo.

|  |  |
| --- | --- |
| CrimesIucrFile | C:\sites\ChicagoCrimesDW\ETL\Sources\iucrFile.csv |
| CrimesCommAreasFile | C:\sites\ChicagoCrimesDW\ETL\Sources\CommAreas.csv |
| CrimesMainFile | C:\sites\ChicagoCrimesDW\ETL\Sources\CrimesFile.csv |
| CrimesSampleFile | C:\sites\ChicagoCrimesDW\ETL\Sources\Sample.csv |
| TimeDimension | C:\sites\ChicagoCrimesDW\ETL\Sources\TimeDimension.csv |

**Execução:**

A execução se dá de maneira completa através do job *ChicagoCrimes.kjb*. No “*Main File Input*” da transformation *main.ktr*, pode-se escolher entre o arquivo de amostra e o arquivo principal, utilizando as variáveis acima. Para uma nova execução do job completo, é importante que as tabelas estejam ***truncadas****.* Pode-se executar usando a tecla F9 ou clicando no botão “*Run*” e em seguida “Run” novamente.

Execuções faseadas podem ser feitas acessando cada *transformation*.

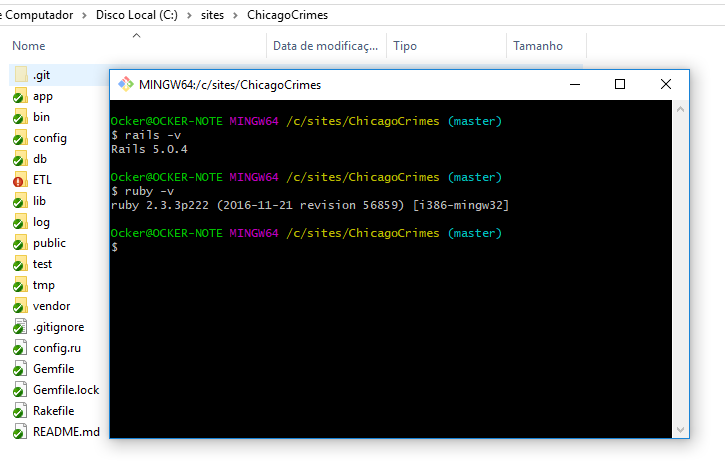
**Em caso de erro**, analisar rapidamente as linhas vermelhas no painel Logging, principais erros são:

* Arquivo não encontrado -> Verificar pasta source, nomenclatura dos arquivos e se as variáveis setadas estão de acordo com o caminho e nomes dos arquivos.
* Conexão com BD -> Verificar se a conexão ao BD está correta de acordo com o print acima, certificar-se de que a conexão esteja acessível em todas as *transformations* (*Shared connection* deve estar em negrito, se não, “*Menu de contexto*” >> “*Share*”).
* Tabelas destino não truncadas -> Executar *Truncate table* nas respectivas tabelas.

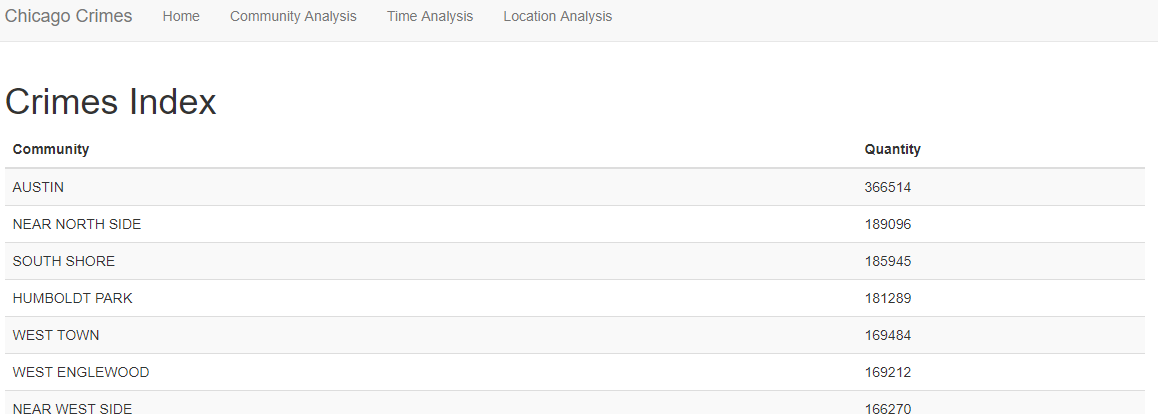
**Aplicação ChicagoCrimesAnalysis:**

**GIT:** <https://github.com/BI-RDChicago/ChicagoCrimesAnalysis>

**Versões do Ruby e Rails:**



**Aplicação:**

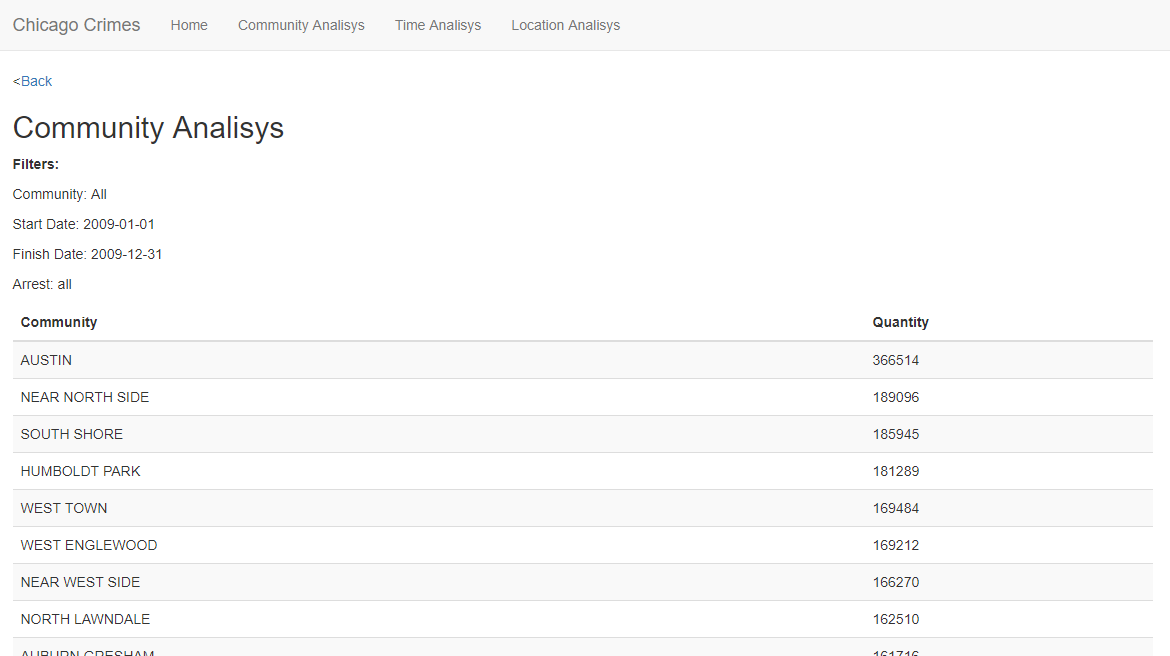


Foram criadas 3 análises básicas para compor o desafio proposto:

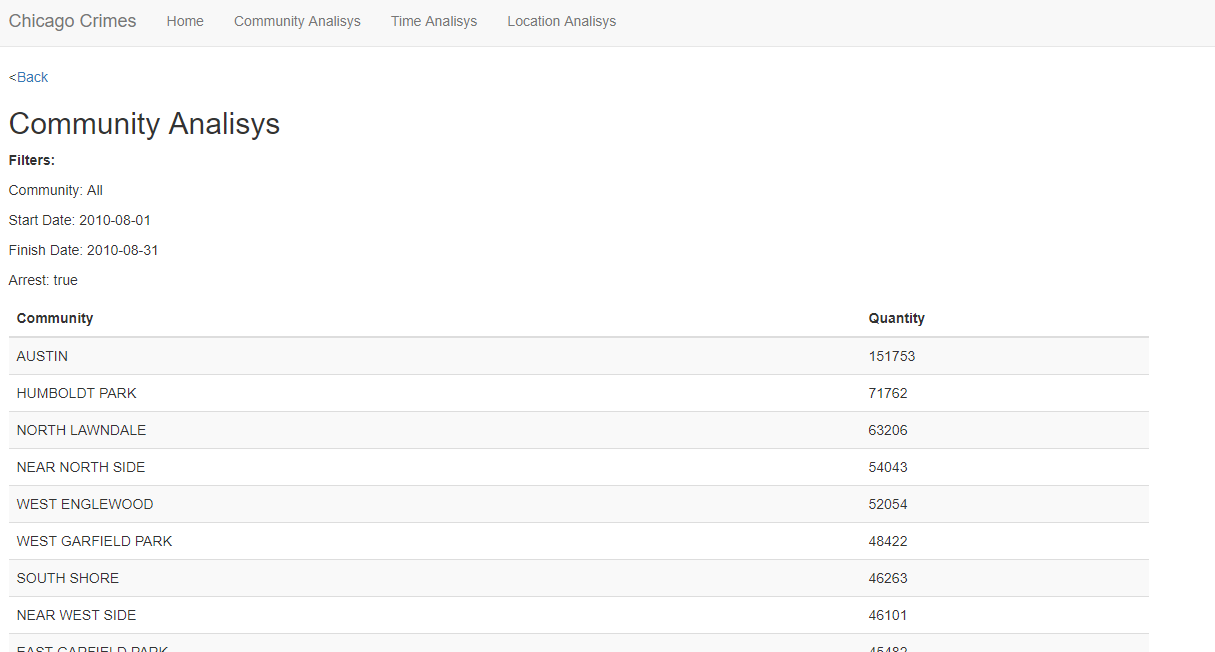
* Community Analysis - Agrupado por Community
* Time Analysis - Agrupado por tempo
* Location Analysis - Agrupado por local do crime

**Perguntas do desafio:**

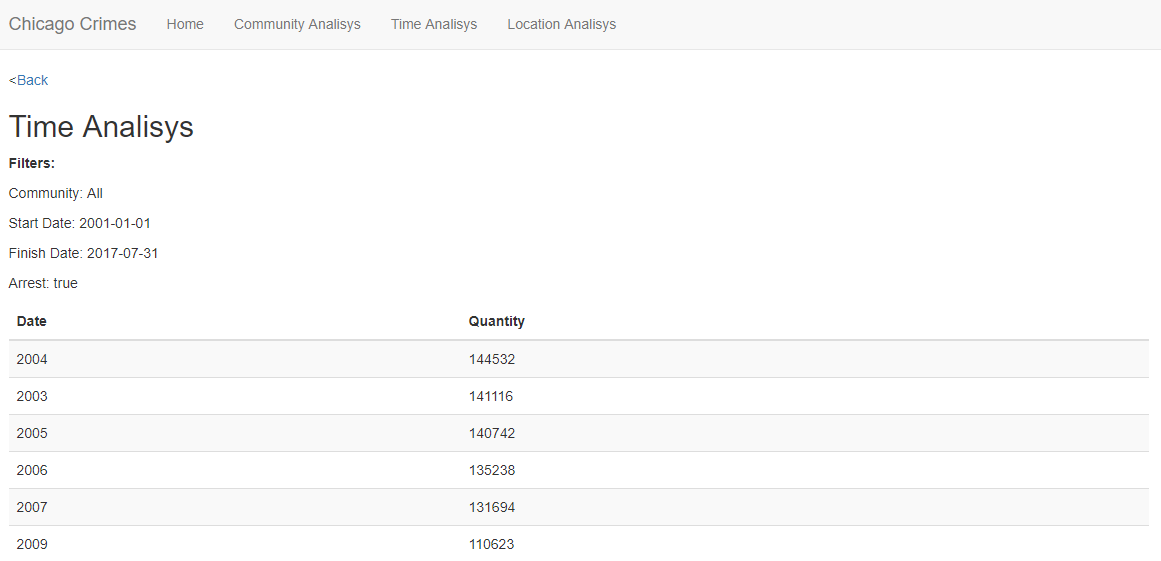
1. Qual **Community Area** teve mais crimes no ano de 2009:



1. Qual **Community Area** gerou mais prisões no mês de Agosto de 2010:

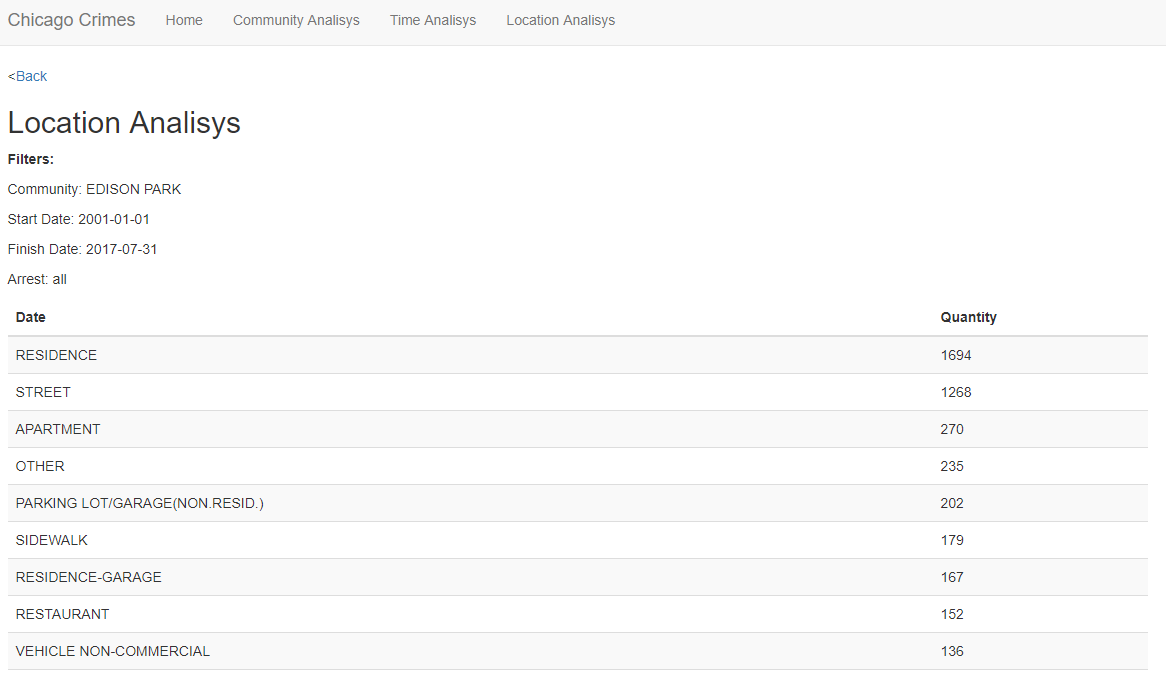


1. Qual foi o ano com mais prisões em Chicago?



1. Crie pelo menos mais uma pergunta que deverá ser respondida pelo seu DW.

* Qual a distribuição, por localização do crime, para **Community Area** com menos crimes no histórico?



1. Fique livre para criar outras análises que você achar interessante.

O item “Location analysis” permite agrupar por local do crime, permitindo ver a distribuição dos crimes nos locais, filtrando por “Community Areas” mais ou menos “perigosas”.

**Melhorias aplicadas:**

O modelo dimensional criado segue os princípios de uma modelagem estrela, com o menor grão de tempo em dias para a tabela fato Crimes.

Foi dada atenção especial a controller, que como foi apontado no documento de pontos a melhorar, agrupava toda a lógica de consultas e controles. Foram criados novos métodos para evitar repetição de código e também as consultas foram tratadas a partir da model Crimes, adequando um pouco melhor ao padrão MVC.

Ainda na *model* Crimes, creio que seja possível melhorar ainda mais o reuso de código a partir de novos métodos que criem *scopes* mais genéricos, usados em mais de uma consulta, permitindo que o *scope* seja especializado conforme a necessidade e melhorando a manutenabilidade.

Novamente a performance do ETL caiu consideravelmente a partir do momento do acoplamento do ETL à base gerada a partir do Rails, creio que pode haver um ganho considerável de performance após um estudo sobre suas causas (index, FKs, etc).

Foram feitas adequações na nomenclatura dos métodos e variáveis, ternários e também na ferramenta usada (Sublime) para seguir as recomendações de estilo apontadas pelo doc de melhorias e pelo *The Ruby Style Guide* do github da RD.

Infelizmente não tive tempo ainda de aplicar GuardClauses e testes unitários através de specs, mas estou estudando.

**Repositório:**

**Git – Branch:**

<https://github.com/BI-RDChicago/ChicagoCrimesDW/tree/Dev>